RMIT UNIVERSITY

Flight Controller Design, Automatic Tuning and Performance Evaluation of Quadrotor UAVs

Xi Chen

School of Electrical and Computer Engineering

College of Science, Engineering and Health

RMIT University

A thesis submitted in fulfilment of the requirements
for the degree of

Doctor of Philosophy

February 2017

I would like to dedicate this thesis to my loving wife, Xiaoyu Li.

Declaration

I certify that except where due acknowledgement has been made, the work is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program; any editorial work, paid or unpaid, carried out by a third party is acknowledged; and, ethics procedures and guidelines have been followed.

(sign here)

(name here) Xi Chen

(date here) 05 February 2017

Acknowledgements

I would like to thank several people who made this research possible. Foremost I would like to express my sincere gratitude to my supervisor, Prof. Liuping Wang for her insights, constant support, helpful advice and paper editing skills throughout the duration of this research. She consistently allowed this paper to be my own work, but steered me in the right the direction whenever she thought I needed it.

I am grateful to my former and current lab mates of Advanced Control Group, especially Ki Chun Ng, Lu Gan, Shan Cai, Robin Guan, Pakorn Poksawat, Yuankang Zhu, Yangbo Liu, He Gong and Dharma Aryani, for the great time spent together and experiences shared.

Last but not the least, many thanks to my family for their patience and the support in all my choices. In particular, I thank my beloved wife, Xiaoyu Li, for her company and understanding during the journey of research.

Thanks for all your encouragement!

Abstract

For the last two decades, with the fast development of electronics, cheap and lightweight flight controllers, accelerometers, global positioning system and cameras have become readily available, which lead to a rapid growth of small commercial multi-rotor Unmanned Aerial Vehicles (UAV). The multi-rotor unmanned aerial vehicles are unstable systems due to their physical structures and feedback control systems underpin their operations and are paramount in the design and implementation of unmanned aerial vehicles. This thesis is dedicated to the topics of quadrotor controller design, automatic tuning and performance evaluation of their feedback control systems. Six key contributions are made in this research: (1) critical evaluation of conventional model-based quadrotor UAV controller design; (2) novel quadrotor PID control automatic tuning process with potential wide applications; (3) quadrotor control system performance assessment method via step response identification; (4) design and performance assessment of cascade and centralized quadrotor control using Model Predictive Control (MPC); (5) design and simulation of Quadrotor Discrete-Time One-step-ahead Predictive Control (DOPC) design; (6) design and implementation of two quadrotor UAV test rigs for validations of the proposed control strategies.

In this thesis, a critical review of the conventional conventional model-based quadrotor flight controller design is firstly conducted. Numerical simulations and experimental tests show the simplified integrator model and the neglected dynamics, which are commonly used in the conventional quadrotor controller design, can dramatically affect the closed-loop responses of the systems. The problems existed in the conventional model-based PID controller design serve the motivation to develop the PID autotuner.

The key idea behind the proposed auto-tuner is to fit the dominant dynamics of a physical plant into a simplified integrator plus delay model, which is then used to design the PID controller. The auto-tuner ensures a sound closed-loop control performance, without endangering the unmanned aerial vehicles and providing engineers and practitioners with reliable controller parameters. Another important issue for future quadrotor applications is how to reliably assess its closed-loop control performance and identify potential faults before flight. To address this problem, a novel approach to assess the closed-loop control performance of a quadrotor UAV is proposed in this thesis, which gives an intuitive way to more precisely evaluate the designed controllers and to obtain better controller tuning parameters. The auto-tuner and the proposed performance assessment scheme are evaluated on the self-made test rigs with final validations on flight tests.

This thesis also covers the topic of quadrotor linear and nonlinear predictive control. For linear MPC, both centralized MPC and cascade MPC control architectures are designed for the quadrotor position controls. From the comparison, the advantages of cascade MPC over centralized MPC are highlighted. In addition to linear MPC, a novel optimization based discrete-time nonlinear predictive controller (DOPC) is developed. The main benefit of DOPC is to enable the quadrotor to fly with large Euler angles, so that it can make aggressive manurer and handle larger disturbance. Robustness analysis is conducted on the DOPC quadrotor control system.

Contents

List of Figures

Nomenclature

Symbols

$\bar{\omega}_m$	Rotor's rated speed
$\boldsymbol{\eta}$	Parameter vector in the Laguerre expansion
$\boldsymbol{\lambda}$	Lagrange multiplier
$\boldsymbol{\Omega}, \boldsymbol{\Psi}$	Pair of matrices in the cost of predictive control
$\boldsymbol{\theta}, \boldsymbol{\Phi}$	Parameter vector and regressor vector in system identification
$\boldsymbol{A}$	State matrix of state-space model
$\boldsymbol{B}$	Input-to-state matrix of state-space model
$\boldsymbol{C}$	State-to-output matrix of state-space model
$\boldsymbol{D}$	Direct feed-through matrix of state-space model
$\boldsymbol{F}(\cdot)$	Nonlinear term in DOPC state-space equation
$\boldsymbol{F}_M, \boldsymbol{\Phi}_M$	Pair of matrices used in the prediction equation $\boldsymbol{Y} = \boldsymbol{F}\boldsymbol{x}(k) + \Phi\Delta\boldsymbol{U}$
$\boldsymbol{I}_{q\times r}$	Identity matrix
$\boldsymbol{K}_y$	State feedback control gain related to y
$\boldsymbol{K}_{mpc}$	Feedback control gain using MPC
$\boldsymbol{L}(\cdot)$	Discrete and continuous-time Laguerre functions

$\boldsymbol{M}, \boldsymbol{\gamma}$ Pair of matrix, vector for inequality constraints $\boldsymbol{M}\boldsymbol{x} \leqslant \boldsymbol{\gamma}$

$\boldsymbol{o}_{q \times r}$ zero matrix

$\boldsymbol{Q}, \boldsymbol{R}$ Pair of weight matrices in the cost function of predictive control

$\boldsymbol{R}_{B \to V}$ Transformation matrix from $\{B\}$ to $\{V\}$

$\boldsymbol{u}(\cdot)$ Control signal

$\boldsymbol{x}(\cdot)$ State variable

$\boldsymbol{y}(\cdot)$ Output signal

$\boldsymbol{Y}$ Predicted output data vector

$\Delta \boldsymbol{U}$ Parameter vector for MPC control sequence

$\Delta \boldsymbol{u}(\cdot)$ Incremental control

Δt Sampling interval

ϵ Relay hysteresis

ϵ_m Actuator's time constant

ϵ_{ss} Desired continuous-time closed-loop time constant

$\hat{\tau}_I$ Normalized integral control time constant

$\hat{K}_c$ Normalized proportional control gain

ω_1 Frequency of the periodic signal in the relay test

ω_d Discrete-time fundamental frequency $\omega_d = \frac{2\pi}{N}$

ω_n Natural frequency

ω_{ds} Discrete-time fundamental sampling frequency $\omega_{ds} = \frac{2\pi}{N_s}$

$\omega_{m_{1,2,3,4}}$ Angular speed of each rotor

$\phi,\ \theta,\ \psi$ Roll, pitch and yaw angles

σ	Noise variance
τ_D	Derivative control constant
τ_I	Integral control time constant
$\tau_{x,y,z}$	Torques on x, y and z body axes
ζ	Damping ratio
$\{B\}$	Body reference frame
$\{I\}$	Inertial reference frame
$\{V\}$	V reference frame
a	Amplitude of the relay
a	Scaling factor for discrete-time Laguerre functions
b_t	Thrust constant
$C(s)$	Controller transfer function
d_m, d	Time delay
$d_{1,2,3,4}$	PWM signal's duty cycle of the ith DC motor drive
d_{mm}	Distance from the motor to the mass center
$e(t_k)$	Discrete-time error signal
F^l	lth frequency sampling filter
f^l	output of the lth frequency sampling filter
g	Gravitational acceleration
$G(s)$	Open-loop transfer function
$I_{xx,yy,zz}$	Moment of inertia about x, y and z body axes
J	Performance index for optimization

K	DC gain of the integrator plus delay transfer function
K_d	Derivative control gain
k_d	Drag constant
K_i	Integral control gain
$K_p,\ K_c$	Proportional control gain
m	Mass
m	Number of inputs, also the mth future sample in discrete time
N	Number of samples within one period
N	Number of terms used in Laguerre function expansion, both continuous and discrete time
n	Number of frequencies included in the Frequency Sampling Filters model
n	Number of state variables
N_c	Control horizon
N_p	Prediction horizon
N_s	Number of samples within the settling time
$p,\ q,\ r$	Body frame angular velocities
q	Number of outputs
q^{-i}	Backward shift operator
r	Reference signal
r_{wv}	Actuator's steady-state gain
$S_x,\ C_x,\ T_x$	$sin(x)$, $cos(x)$ and $tan(x)$ shorthand
T	Total thrust

Nomenclature

t Time

$T(s),\ T(z)$ Closed-loop transfer function

T_p Period of the oscillation in the relay test

T_s Settling time

$T_{1,2,3,4}$ Thrust produced by each rotor

u Control signal

$u_{x,y,z,T}$ Quadrotor desired control signals

$v(k)$ output measurement noise

$v_{1,2,3,4}$ Armature voltage of the ith motor

V_{bat} Battery voltage

$x,\ y,\ z$ Positions in the inertia reference frame

$x^V,\ y^V,\ z^V$ Positions in the V reference frame

Abbreviations

DOPC Discrete-Time One-step Ahead Prediction Control

I-DOPC Integral discrete-time one-step-ahead predictive control

IMU Inertial measurement unit

MCU Microcontroller unit

MIMO Multiple-input, multiple-output

MPC Model Predictive Control

PPM Pulse Position Modulation

PWM Pulse Width Modulation

QP Quadratic Programming

RC Radio control

SISO Single-input, single-output

UAV Unmanned aerial vehicle

Chapter 1

Introduction

1.1 Motivations and Objectives

Over the last two decades, the Unmanned Aerial Vehicles (UAV) have drawn significant attention in both civilian and military aspects, due to that they have high potentials in Intelligence, Surveillance, and Reconnaissance (ISR) applications with less cost and more flexibility than manned aircraft. A quadrotor, also called a quadcopter, is a multirotor helicopter lifted and propelled by four rotors. It is an unique type of UAV which has Vertical Take Off and Landing (VTOL) ability. Figure 1.1 shows quadrotor's classification in different kinds of aircraft [Bouabdallah, 2007]. The quadrotor architecture has become a very popular one because of its simple mechanical structure and favorable dynamic characteristics. Although quadrotors are open-loop unstable, like most rotorcraft architectures, they exhibit a good degree of decoupling, which makes the flight controller design easier than conventional helicopters [Bergamasco and Lovera, 2014]. Additionally, with the fast development of electronics industry, nowadays most of the commercial quadrotor platforms are equipped with electronic stabilization systems at an affordable price.

A quadrotor UAV is an unstable system because of its physical structure, thus, control system design and implementation are paramount in the applications of quadrotor UAVs[Li and Song, 2012]. Many control methods have been proposed for the quadrotor attitude control problem, such as PID and linear quadratic control [Bouabdallah et al., 2004; Corke, 2011], MPC control [Alexis et al., 2010], H_∞ control [Raffo et al.,

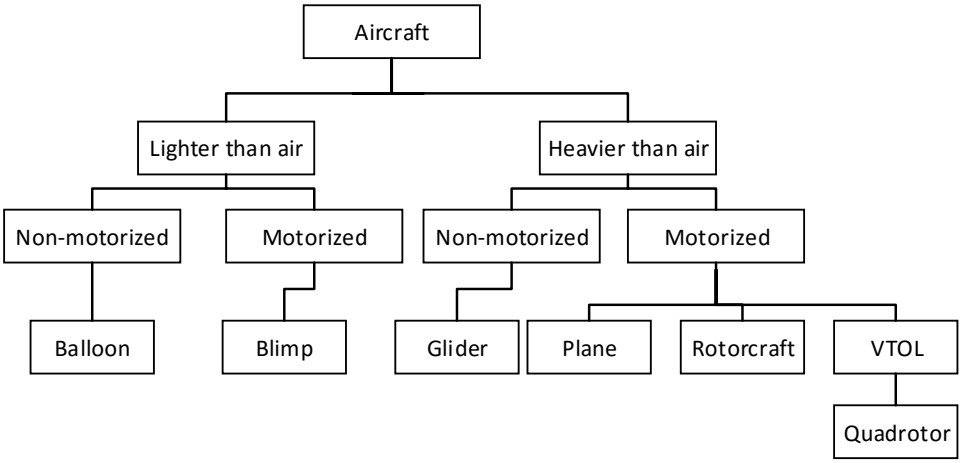


Figure 1.1: Aircraft classification

2010], backstepping control [Ha et al., 2014; Madani and Benallegue, 2006] and sliding mode control [Lee et al., 2009]. However, Bouabdallah [2007] and Pounds et al. [2010] discussed that, in practice, cascaded PID control, which is also the most widely used quadrotor control technique, provides comparable or even better performance than more complex controllers due to its simplicity and robustness.

The quadrotor PID flight controller design normally consists the following steps. The first step is to develop the mathematical model of the quadrotor from physical laws. The second step is to measure the physical parameters. The third step is model-based flight controller design and simulation.

One of the problems in the conventionally model-based quadrotor controller design is, the commonly used integrator model can be overly simplified. More specifically, in the cases when the actuator dynamics is not sufficiently fast, neglecting the actuator dynamics may seriously degrade the closed-loop system's performance. Using a high order model for controller design could solve this problem. However, high order model may complicate the design process, and more importantly, model parameter mismatches can still degrade the closed-loop system's performance. To obtain a quadrotor's mathematical model, physical parameters of all quadrotor components, such inertia terms, the thrust constant and the drag constant need to be measured. The errors of these parameters may accumulate, resulting in an inaccurate model. Considering the quickly expanding quadrotor hobby market, where the accurate physical

parameters of the quadrotor are rarely provided by manufacturers and hard to measure, developing an automatic tuning procedure of the quadrotor PID controller is both necessary and significant. This auto-tuner should lead to the suitable PID control parameters with out any human intervention.

Aside from control system design and implementations, another important issue for future quadrotor applications is how to reliably assess its closed-loop control performance without endangering the UAV. Possibly, it also has implications on faulty detection and avoidance in the early stage.

In order to assess the closed-loop control performance of a UAV, it is well understood that the numerical simulation[Lee et al., 2009; Lungu and Lungu, 2013; Mian and Wang, 2008; Voos, 2009] and dynamic response tests are required to reveal whether the closed-loop system has a suitable settling time, under-damped modes, and steady-state error. However, it will be discussed later in this thesis, the numerical simulations may not reveal the true dynamics of the system due to the inaccurate plant parameters measured. Additionally, the direct step response testing results can also be masked and corrupted by noise and disturbances. Thus, one of the objectives of this research is to find another way to more precisely evaluate the designed controllers and to obtain better controller tuning parameters.

Quadrotors are nonliner multi-input and multi-output systems. In order to design PID controllers for a quadrotor, the nonlinear equations need to be linearized under the small Euler angle assumption. Additionally, the linearized system still requires decoupling before PID controllers can be designed for each channel separately. A few problems exist in such design method. Firstly, in order to maintain the small Euler angle assumption, the Euler angle reference signals need to be limited. However, in a cascade PID control structure, those Euler angle reference signals are generated by three individual controllers so they are not optimized. Secondly, constraints need to be applied on the control signals to protect the actuators. Nevertheless, for the same reason, the control signals are not optimized in the cascade PID control structure. There are two possible ways to solve above problems. The first one is to employ the linear MPC to deal with the constraints on the Euler angle reference signals and the control signals. The second one is to use a nonlinear control technique that can optimize the quadrotor's control signals and make it work under any Euler angles. What has been discussed above becomes the motivations of design linear MPC and

nonlinear predictive control on the quadrotor.

1.2 Literature Review

1.2.1 Quadrotor UAVs

In late 2000s, with fast development of electronics, cheap and lightweight flight controllers, accelerometers, global positioning system and cameras became readily available, leading to a rapid growth of small commercial quadrotors along with other multirotor configurations [Gupte et al., 2012; Lucieer et al., 2014; Sarris and ATLAS, 2001]. In the meantime, many research groups began to use quadrotors as their UAV research platforms. For example, In 2010, the University of Tasmania and the Australian Antarctic Division made use of Mikrokopter (an open source quadrotor plateform) to monitor moss beds in Antarctica [Lucieer et al., 2012]. The same research group also used a quadrotor UAV for forest change detection purpose [Wallace et al., 2012]. There are also a few university-level quadrotor UAV projects. In [Sa and Corke, 2011, 2012] , the author proposed to use low-rate onboard laser range finder instead of onboard cameras to realize the velocity estimation. The test bed for their research is MikroKopter research platform. In [Singh and Fuller, 2001], the research group focus on the development of trajectories and controllers enabling aggressive maneuvers such as flying through narrow, vertical gaps and perching on inverted surfaces with high precision and repeatability. Many research groups or institutions have constructed their own quadrotors to suit specific purposes [Bouabdallah and Siegwart, 2006; Guenard et al., 2008; Hoffmann et al., 2004; Meier et al., 2011]. There are also many commercially available platforms such as Draganflyer X4, Asctec Hummingbird, Gaui Quad flyer, Parrot ARDrone, and DJI Wookong. Additionally, there are a number of open source projects for quadrotors such as Arducopter, Openpilot, Paparazzi, Pixhawk, Mikrokopter, KKmulticopter, Aeroquad. Those test platforms make the verification of the proposed control methods much easier.

Quadrotor trajectory planning and tracking is another widely studied topic. In [Hoffmann et al., 2008], the author developed an approach to track a planned sequence of desired trajectory. In [Teuliere et al., 2011], the problem of chasing a moving ground target is analyzed. In [Alexis et al., 2012], the impact of wind disturbance in quadrotor

trajectory tracking is analyzed and the wind gust model is also given. In [Escareño et al., 2013], the author studied the trajectory control problem of a quadrotor subject to 2D wind disturbances. In [Bouktir et al., 2008], the author studied optimal trajectory planning problem.

1.2.2 Automatic tuning of PID control

Automatic tuning of PID control has been the pursuit of a generation of control engineers for the past several decades [Åström and Hägglund, 1984, 2006; Chen and Wang, 2016; Hagglund and Astrom, 1985; Hang and Sin, 1991; Johnson and Moradi, 2005; Yu, 2006]. With the functionality of automatic tuning, a PID controller will have the capability of setting its parameters with minimal human intervention.

There are several books dedicated to the topics of PID control system design and auto-tuning of this type of controllers [Åström and Hagglund, 1988; Åström and Hägglund, 2006; Johnson and Moradi, 2005; Yu, 2006]. A tutorial overview of auto-tuning using relay feedback control is given in [Hang et al., 2002].

It is well understood that a key component among all auto-tuners is the application of system identification tools to find the mathematical representation of the physical systems. There are many papers published in the control engineering literatures for automatic tuning of PID controllers. The following discussions only give a glimpse of the vast literatures. Hang and Sin [1991] suggested the use of a set of pseudo-random-binary-sequence (PRBS) signals together with cross-correlation technique to identify the critical information needed for auto-tuner design. Wang et al. [2001] proposed to use the Fast Fourier Transform (FFT) to obtain the plant frequency response, the step response and a second order model with time delay for automatic tuning of PID controllers. Kaya [2003] used relay feedback control to generate identification data for obtaining a transfer function model with delay by utilizing the information of peak amplitude and frequency where the delay was compensated using a modified PI-PD Smith predictor. Jeng and Lee [2012] and Jeng [2014] proposed an automatic tuning algorithm for cascade control systems using a step input test signal with the estimation of approximate models for the primary and secondary plants in which the internal model control design method was deployed. Johnson and Moradi [2005] and Yu [2006] have introduced a range of PID controller design methods, including system identifica-

tion using relay-feedback experiments and PID controller design methods. Lim et al. [2012] applied recursive least squares to estimate a discrete-time model to improve the closed-loop control performance of a sheet metal forming process. The approach used in [Gyöngy and Clarke, 2006] was to obtain a design-point on the Nyquist diagram by injecting a sinusoidal signal for identification. Romero et al. [2011] proposed an auto-tuning algorithm based on relay feedback control experiments by minimizing the load disturbance effects and maximizing the integral gain with minimum constraints on the gain and phase margins. Yu Jin et al. [2014] extended PID auto-tuning method to applications in the area of fraction order plus time delay system. Dittmar et al. [2012] proposed the use of nonlinear optimization techniques to find the PID controller parameters in a multi-loop controller tuner. Ho et al. [2000] extended the gain and phase margin design for PID control of single-input and single-output systems to multi-input and multi-output systems. Cetin and Iplikci [2015] proposed an auto-tuner for nonlinear and multivariable systems. There are also a number of modelling and auto-tuning methods based on step responses. In [Bi et al., 1999], the author proposes a simple yet robust identification method for a linear monotonic process, derived from a step test. In [Huang et al., 2001], a system of procedures for identification using the transient step response was presented. In [Wang and Cluett, 1997a], the author developed a methodology for step response identification using the frequency-sampling filter (FSF) model.

Over the years of research, consensus has been reached that the mechanism of relay feedback control, used byÅström and Hägglund [1984], is the most effective instrument that has been widely used in the design of subsequent auto-tuners. There are three key reasons for the relay feedback control applications in the design of auto-tuners. The first is that the relay feedback control will automatically generate an excitation input signal to ensure successful identification of process information that is relevant for the PID controller design. This is achieved without a priori information about the system dynamics, which is paramount in the design of an automatic tuning algorithm. The second is that the system is under closed-loop control with the nonlinear relay controller so that it is maintained at the operating condition chosen by the user. The third is that the sustained periodic oscillation generated by the relay feedback control produces an excellent signal-to-noise ratio for the estimation of critical process frequency information, which can be directly extracted using Describing Function Analysis [Åström

and Hägglund, 1984; Atherton, 1975], Fast Fourier Analysis [Wang et al., 2001] and frequency sampling filter based estimation [Wang and Cluett, 2000].

However, the cost of the periodic oscillation when using the relay feedback control is that the other process information, excluding the critical frequency information, is not directly available. Because of this, the earlier generations of auto-tuners have focused on PID controller design using the critical frequency response information [Åström and Hägglund, 1984; Åström and Hagglund, 1988; Ho et al., 1995; Wang et al., 1995]. Gradually, it was recognized that the PID controller design using one critical frequency response information has limitations on the achievable performance [Åström and Hägglund, 2006]. Perhaps, one of the ways forward is to convert the frequency information obtained from the relay feedback control experiment to a parametric model [Hang et al., 2002; Johnson and Moradi, 2005; Lim et al., 2012]. Once a parametric model is obtained, the PID control design methods, such as Internal Model Control (IMC) design [Rivera et al., 1986], can be readily used.

1.2.3 Model Predictive Control

Model Predictive Control (MPC) refers to a class of algorithms that compute the trajectory of manipulated variable adjustment to optimise the future behaviour of a system. Since the 1970s, this class of algorithms has been under continuous development and has achieved wide applications in the process industries. The comprehensive reviews of MPC can be found in several books [Allgöwer and Zheng, 2012; Camacho and Bordons, 2013; Grüne and Pannek, 2011; Kwon and Han, 2006; Rossiter, 2013; Wang, 2009] and papers [Bemporad and Morari, 1999; Findeisen and Imsland, 2003; Mayne et al., 2000; Morari and H. Lee, 1999; Qin and Badgwell, 2000, 2003].

The initial ideas of receding horizon control and model predictive control can be traced back to 1960s [Garcia et al., 1989]. In 1980s, the publications of the paper on IDCOM [Richalet et al., 1978], Dynamic Matrix Control (DMC) [Cutler and Ramaker, 1980], and Generalized Predictive Control (GPC)[Clarke et al., 1987a,b] attracted more researchers' interest in this area.

DMC and GPC were developed for different purposes. DMC was firstly invented and applied in oil and chemical industries to solve the constrained MIMO system control problem. GPC was studied as a new adaptive control method. However, the GPC's

formulation makes it unsuitable for MIMO constrained systems, which are mostly the case for chemical industries. DMC has exerted a great impact on the chemical industry and initial research of MPC began with the attempts to understand DMC by chemical engineers. Due to the simplicity of the algorithm and the use of step response model, MPC quickly became popular in process industries. However, the algorithms developed during that period for process industries lacked solid theoretical proof of feasibility, stability and robustness.

Formal theoretical analysis of MPC was then conducted under state-space formulation[Morari, 1994]. Under the framework of the state-space model, known theorems of state space theory can be used as tools to analyze more complex MIMO systems, noninear plants and systems subject to unknown external disturbances.

The stability of constrained receding horizon controllers [Rawlings and Muske, 1993; Rossiter and Kouvaritakis, 1993] was another challenging problem to be tackled. It was shown in above papers that optimality does not imply stability. Terminal penalties, Lyapunov functions or invariant sets were employed to guarantee the stability of the system. Different techniques to address the stability problem of constrained MPC are summarized in [Mayne et al., 2000], where general sufficient conditions to design a stable constrained MPC were presented.

For practical applications, one essential question of MPC is its robustness to model uncertainty and noise. The robustness of a system means its capability to maintain stable under a certain range of model parameter variation and noise signals. Various robust MPC design methods can be categorized in two groups. The first one is the indirect way by specifying the performance objective and uncertainty description in a way that the optimal control computations lead to robust stability. The second one is the direct way by enforcing a type of robust contraction constraint which guarantees that the state will shrink for all plants in the uncertainty set [Bemporad and Morari, 1999].

Even though the majority of MPC applications are based on the linear model, there are cases where nonlinear factors are so significant that nonlinear MPC design becomes necessary. Examples of applications of nonlinear MPC include pH control and polymer manufacturing in the chemical industry [Qin and Badgwell, 2000]. Initial attempts to deal with nonlinearity appeared in a few early MPC papers [Prett and Gillette, 1980; Richalet et al., 1978]. Theoretical analysis nonlinear MPC have in given in several

publications [Diehl et al., 2009; Lalo Magni et al., 2009; Mayne, 1997; Meadows and Rawlings, 1997]. Nonlinear model development, state estimation and rapid solution of the control algorithm in real time are recognized as the three main obstacles of nonlinear applications.

1.2.4 Nonlinear model predictive control and Finite Control Set

In last several years, the Finite Control Set (FCS)-Model predictive control (MPC) has become popular in the research field of power electronics and motor drive control, due to its simple concept and robust performance. In FCS-MPC, The term finite control set (FCS) is given because for a two level voltage source inverter, there are eight combinations of inverter states and thus the there are only eight possible quantized control signals. The optimization of the inverter states is performed using the receding horizon control principle, which is the core of model predictive control. FCS-MPC is termed by combining the finite control set and the model predictive control technology. One of the early works of FCS was presented in [Kükrer, 1996]. Similar control techniques have also been published with different applications, such as permanent magnet synchronous motors [Moon et al., 2003; Morel et al., 2009] and power converters [Rodriguez et al., 2005; Vargas et al., 2009]. More recently, the FCS-MPC algorithms have been reviewed in [Rodriguez and Cortes, 2012; Wang et al., 2015].

In traditional Model Predictive Controller (MPC) design, long control and prediction horizon are needed to ensure the closed-loop stability [Wang, 2009]. However, long control horizon increases the sizes of matrices involved in the on-line computation, which leads to heavy computational burden of the MPC controller, especially for nonlinear systems. One of the most attractive features of FCS-MPC is its one-step ahead prediction technique, which effectively reduces prediction and control horizon to one and thus minimizes the computational requirement. Furthermore, because it is also an optimization based control law, it inherits MPC's feature of dealing with constraints.

The original FCS-MPC did not include the integral action and therefore relied on high gain feedback control to reduce the steady-state errors. In [Wang and Gan, 2014], the authors proposed that an integrator can be used as the outer-loop controller to eliminate the steady-state errors. It is also shown in the same paper that by using

one-step ahead prediction and optimization without constraints, the FCS-MPC system has all eigenvalues on the origin of the complex plane, which is effectively a deadbeat control. Because of the high controller gain, the closed-loop system is robust and less sensitive to system parameter variation.

1.3 Contributions

The section summarizes the six key contributions made in this research:

- critical evaluations of conventional model-based quadrotor controller design;
- a novel quadrotor PID control automatic tuning process;
- a novel quadrotor control system performance assessment method via step response identification;
- design of cascade and centralized quadrotor MPC control systems and critical evaluations of their performance;
- a novel Quardrotor Discrete-Time One-step-ahead Predictive Control (DOPC) design;
- design and implementation of two test rigs for the experimental validation of the proposed auto-tuner and control system design methods.

1.3.1 Conventional model-based quadrotor controller design review

This thesis provides a through review of the conventional model-based quadrotor flight controller design, and finds the major problem of this design methodology. In the conventional way, a simplified integrator model is used for both quadrotor inner-loop and outer-loop control design. However, as it will be discussed in Chapter 3, the ignored dynamics dramatically change the closed-loop characteristics of the systems via frequency response analysis. In other words, the actual closed-loop dynamics can significantly differ from the expected ones, when the actuators or inner-loop dynamics are neglected. As a result, when using the simplified integrator model, the initial controller

parameters calculated by model-based design method can hardly fulfil the control requirements. Those initial parameters have to be carefully adjusted. The tuning process is quite time-consuming and the tuning results largely reply on the control engineer's experience.

1.3.2 Quadrotor PID control automatic tuning process

To solve the problems existed in the conventional model-based quadrotor flight controller design, a novel PID control auto-tuner is developed. Because the integrating systems are on the boundary of instability, they require stabilization before a relay experiment can be safely conducted. It appears that the majority of the auto-tuners are derived for stable systems only [Hang et al., 2002; Johnson and Moradi, 2005] and the use of stabilizing controllers on integrating systems is seldom considered. This thesis brings together several existing approaches in the areas of system identification and PID controller design to form a new auto-tuner for integrating systems. The auto-tuner has four key components that differentiate it from the other auto-tuners. The first is that the relay experiment is conducted for an integrating system under closed-loop control, for the reason that the open-loop system is unstable; the second is that frequency sampling filters are used to recursively estimate the fundamental frequency of the closed-loop system [Wang and Cluett, 2000; Wang et al., 1999]; the third is the simple computation of the frequency response of the integrating system and the associated dynamic model; and the fourth is the PID controller design using the tuning rules for integrator with delay systems presented in [Wang and Cluett, 1997b, 2000]. Because the PID controller design method requires the user to specify a desired closed-loop time constant as the performance parameter, the auto-tuner designed is then naturally suited for tuning a cascade PID control system where the inner-loop control system typically has a much faster dynamic response speed than the outer-loop response, which can be readily achieved by adjusting the two desired time constants.

In terms of the quadrotor flight controller design, the proposed auto-tuner has two advantages over the traditional model-based PID design. Firstly, the tuning process is fully automatic and thus physical parameter measurements are not required. Secondly, the auto-tuner can fit the critical dynamic information of the system into a simple integrator plus delay model. The integrator plus model contains enough information

to determine the PID controller parameters, so it offers better tuning results than using the traditional integrator only model. Compared with the high order model, the auto-tuner's tuning rules are simpler and more reliable, because they are independent of the physical parameter measurements.

1.3.3 Quadrotor control system performance assessment via step response identification

A novel approach to assess the closed-loop control performance of a quadrotor UAV is proposed. An experimental apparatus is designed so that a quadrotor can be operated safely on ground, yet it mimics its flying conditions without endangering its electronics and mechanics. Relay feedback control is used to generate the excitation signals [Åström and Hägglund, 1984; Wang et al., 1999], which will ensure that the frequency contents of the input signals yield meaningful assessment results in addition to easiness and safety for the experimental test. Upon obtaining the closed-loop experimental data, the step responses of a cascade control system are estimated by extracting the key frequency information with a frequency sampling filter model [Wang et al., 1999]. The proposed approach provides engineers with an intuitive way to more precisely evaluate the designed controllers and obtain better controller parameters.

The identified step responses are adequate only in the case of zero quadrotor velocities since the aerodynamic forces and torques are not taken into account during the identification. However, the ground performance assessment is the first step and foremost important step towards performance assessment because if the quadrotor fails this important test, it is unlikely to fly successfully. The assessment here also provides an instrument for early fault detection.

1.3.4 Cascade and centralized quadrotor MPC control comparison

The cascade and centralized MPC control are designed for quadrotors. Even though it is not the first time MPC is utilized as the quadrotor controller, the through comparison of centralized MPC and cascade MPC quadrotor controllers is new. The key advantage of cascade MPC over centralized MPC in quadrotor control problem is all Euler angles

references can be limited within a narrow range so that the operation conditions of the linearized MPC controller can be maintained. Furthermore, as the quadrotor's inner-loop and outer-loop dynamics possess different time constants, tuning MPC controllers for a cascade system is much easier than that in the centralized MPC design.

1.3.5 Quardrotor Discrete-Time One-step-ahead Predictive Control (DOPC)

A novel optimization-based nonlinear control technique called DOPC is developed from FCS-MPC and applied on several electromechanical systems, including quadrotor UAVs. The design of DOPC overcomes three problems of FCS-MPC. Firstly, the FCS-MPC controller was designed based on the linearised model, so its potential of dealing with nonlinearity was explored. In DOPC, the framework of one-step ahead prediction and optimization in [Wang et al., 2015] is extended to nonlinear systems. Secondly, in FCS-MPC, there are no weighting matrices used on the cost function and as a result the controller has a very high gain. Although adding saturations can limit the control signals, the solution is not optimized and the system's control performance may degrade. In DOPC, weighting matrices are added on both output signals and control signals. It will be shown that the closed-loop dynamics can be modified by tuning the weighing matrices. By adding a weighting matrix on the control signal, the closed-loop system poles are not on the origin of the complex plain so the control gain and control signals are reduced. Thirdly, in the original FCS-MPC frame work, there were only a limited number of possible control signals, e.g. eight combinations of inverter states for a two level voltage source inverter. To handle the control signal constraints, the desired control signal are compared with all possible control signals and one of the eight possible control signals, which is closest to the desired control signal, is chosen to be the actual control signal. However, if the control signal is continuous instead of a limited number of states, how to handle constraints remains a problem. Although so far the FCS-MPC technique is mainly used in the field of power electronics, the one-step ahead prediction and optimization frame work can be applied to a wider range of dynamic systems. For the majority of dynamic systems, the control signal is continuous instead of a limited number of states, so the finite control set (FCS) is not adopted in DOPC. Since the cost function J in one-step ahead prediction and optimization is

quadratic and the constraints are linear inequalities, the problem of finding an optimal predictive control becomes finding an optimal solution of a standard quadratic programming problem. A simple algorithm called Hildreth's Quadratic Programming is employed in DOPC to solve it [Hildreth, 1957].

Comparing with other nonlienar control techniques such as sliding mode control [Utkin et al., 2009; Young et al., 1999], feedback linearisation [Khalil and Grizzle, 2002], backstepping control [Khalil and Grizzle, 2002], the most important feature of the DOPC controller is that it is an optimization based control technique and hence quadratic programming can be used to get the sub-optimal solution of controller signals with constraints. Although Nonlinear MPC can also optimally deal with control signal constraints, it generally requires a large amount of computation resource [Diehl et al., 2009; Michalska and Mayne, 1993].

1.4 Publication

Conference Paper

- Chen, X. & Wang, L., PID Control Auto-tuner Design and Validation for Integrating Systems, in IFAC 2017 World Congress, 2017.(submitted)
- Chen, X. & Wang, L., 2016. Discrete-Time One-Step Ahead Prediction Control (DOPC) of a Quadcopter UAV with Constraints. In Australian Control Conference (AUCC). Newcastle.
- Chen, X. & Wang, L., 2016. Quadrotor Cascade PID Controller Automatic Tuning. In Australian Control Conference (AUCC). Newcastle.
- Chen, X. & Wang, L., 2015. Step Response Identification of a Quadcopter UAV Using Frequency-sampling Filters. In The 17thm IFAC Symposium on System Identification. Beijing.
- Chen, X. & Wang, L., 2013. Cascaded model predictive control of a quadrotor UAV. In Australian Control Conference (AUCC). pp. 354-359.

1.5 Outline of Thesis

The remainder of the thesis is arranged as follows. Chapter 2 is an introduction to the quadrotor dynamics and hardware design. The quadrotor's rotational dynamics, actuator dynamics and translational dynamics are explained. Additionally, the hardware design of two quadrotor test-beds, TC1.0 and TC3.0, is presented. Experiments are conducted to measure the physical parameters of the quadrotor TC1.0. These two quadrotor test-beds and their physical parameters are used as a basis to test different control and assessment methods.

In Chapter 3, the traditional model-based PI control design of the quadrotor is discussed. The linearized model and transfer functions of the quadrotor are given first. Secondly, the quadrotor flight controllers are designed based on the simplified integrator model. Closed-loop frequency response tests and step response tests are performed to analyze the effects of the neglected dynamics. Thirdly, the hardware implementation of the quadrotor closed-loop control system is described. At the end of this chapter, TC1.0's flying tests are carried out to test the overall performance of the closed-loop system.

Chapter 4 presents a simple and effective PID controller auto-tuning process. This chapter begins with the introduction of the auto-tuning steps. After that, numerical simulations are performed to validate each stage of this auto-tuning process. In the end, the proposed auto-tuner is applied on the quadrotor TC1.0 and TC3.0 to tune their flight controllers. The stability margins of different PID control structures are compared. Flying test results are also given to further validate the proposed auto-tuning method.

Chapter 5 presents an innovative approach to identify a quadrotor's 3-axis closed-loop step responses. The two steps in this evaluation process are introduced first: data acquisition using relay feedback experiment and the step response identification using frequency sampling filters. After that, comparisons are made among the proposed model identification method, the numerical simulation method and the direct step response experiment. In the end of this chapter, experimental validation of the proposed assessment approach is performed on the quadrotor test-beds.

Chapter 3 and Chapter 4 of the thesis mainly focus on the PI design technique, where the quadrotor system is decoupled into three channels and PID controllers are

designed accordingly. A drawback of such design method is that the control signals subject to constraints are not optimized. In Chapter 6, two different MPC control structures are employed to tackle this problem. In the centralized MPC design, the quadrotor's translational dynamics and rotational dynamics are represented in one set of state-space equations. One centralized MPC controller is designed to control all output signals and handle only the constraints on manipulated signals. In cascade MPC design, the quadrotor system is divided into two subsystems, and two MPC controllers are designed accordingly. Cascade MPC and centralized MPC are compared under the same quadrotor hovering scenario. Numerical simulations are also performed on the quadrotor position loop to analyze how constraints or actuator saturations affect the control performance.

In the previous chapters, both MPC and PID designs are based on the linearized quadrotor model under the small Euler angle assumption. However, under large external disturbances the quadrotor's Euler angles may become large, which means operational conditions of the linear systems are changed. In such circumstance the linear controllers' performance may degrade. In Chapter 7, a novel optimization based discrete-time nonlinear predictive controller called DOPC is presented to solve this problem. The DOPC controller is originated from FCS-MPC but modified to deal with nonlinear control problems. DOPC's tracking performance, robustness are analysed. Simulations of DOPC control are conducted on the quadrotor and other nonlinear plants.

Finally, Chapter 8 concludes the thesis by summarizing the major contributions and proposing future research opportunities.

Three quadrotors are used throughout this thesis for both hardware experiments and numerical simulations. It should be emphasized that TC1.0 and TC3.0 are quadrotor hardware test-beds developed in this research. TC3.0 is a numerical quadrotor model used only for simulations.

Chapter 2

Quadrotor Hardware Design and Dynamics

In this chapter, firstly the quadrotor's dynamics and hardware design are described. The quadrotor's dynamics include the rotational dynamics, actuator dynamics and translational dynamics. The quadrotor's rotational and translational dynamics are derived from Euler's equations and Newton's second law so they are independent of the quadrotor hardware. The quadrotor actuator's dynamics are depending on the actuator hardware so the actuator equations derived in this chapter are accurate only for this specific quadrotor platforms. Nevertheless, the method of analyzing the effects of the actuator dynamics can be applied to other multirotor platforms. Additionally the hardware design of two quadrotor platforms TC1.0 and TC3.0 are presented. Experiments are conducted to measure the physical parameters of TC1.0.

2.1 Quadrotor Dynamics

This section is an introduction of quadrotor's translational and rotational dynamics, as well as the actuator dynamics. When flying, the quadrotor's four actuators(propellers) spin at the same or different speeds to generate the lift thrust and torques. When the two propellers on the same arm spin at different speeds, the torque on that axis is not zero, which results in a change to the corresponding body angular velocity and Euler angle. The changes in the Euler angles generate horizontal force components, which

control the horizontal motion of the quadrotor. The vertical motion of the quadrotor is controlled by the total thrust generator by four propellers.

2.1.1 Quadrotor Rotational Dynamics

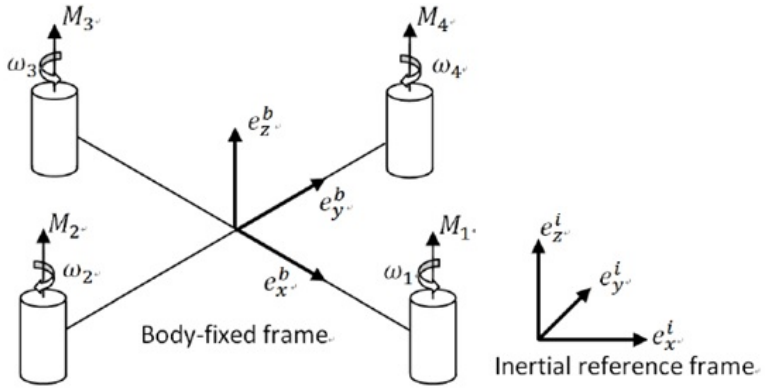


Figure 2.1: Quadrotor reference frames

Figure 2.1 illustrates the framework used to find a quadrotor dynamic model [Bouabdallah et al., 2004; Corke, 2011; Derafa et al., 2006], where M_1, M_2, M_3 and M_4 represent the four rotors. The body reference frame $\{B\}$ is attached to the quadrotor body, with its origin on the mass center of the quadrotor and z axis upwards. The x-axis of $\{B\}$ points to M_1 and the y-axis points to M_4. The inertia reference frame $\{I\}$ is fixed on the ground. The quadrotor's attitude is defined by three Euler angles in $\{B\}$, namely roll (ϕ) about $x-$axis, pitch (θ) about $y-$axis, and yaw (ψ) about $z-$axis. The transformation sequence is $\psi \rightarrow \theta \rightarrow \phi$ in order to obtain the unique solution. The quadrotor rotational dynamics is described by Euler's equation of motion[Bouabdallah et al., 2004; Corke, 2011; Derafa et al., 2006]:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} (I_{yy} - I_{zz})qr/I_{xx} \\ (I_{zz} - I_{xx})pr/I_{yy} \\ (I_{xx} - I_{yy})pq/I_{zz} \end{bmatrix} + \begin{bmatrix} 1/I_{xx} & 0 & 0 \\ 0 & 1/I_{yy} & 0 \\ 0 & 0 & 1/I_{zz} \end{bmatrix} \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix}, \tag{2.1}$$

where I_{xx}, I_{yy} and I_{zz} are the moments of inertia for the three axes in x, y, z directions, p, q and r are their angular velocities and τ_x, τ_y, τ_z are the corresponding torques. Furthermore, the relationship between Euler angular velocities and the body frame

angular velocities (p, q and r) are described in the following differential equation:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi)/\cos(\theta) & \cos(\phi)/cos(\theta) \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{2.2}$$

In order to fly a UAV, the closed-loop control of the three Euler angles ϕ, θ and ψ is necessary. Therefore, there are three outputs in the control system. The manipulated variables or the control signals are the three torques, τ_x, τ_y, τ_z, along the x, y and z directions. The body frame angular velocities p, q and r along the x, y and z directions are the intermittent variables in the cascade control system which is to be configured in the sequel.

The quadrotor is assumed to have symmetric structure with four arms aligned with the $x-$axis and $y-$axis, and as a result there is no interaction between the torques along the three axes. Because of the products of angular velocities in (2.1) and the sinusoidal functions in (2.2), the quadrotor system is a nonlinear system.

When the quadrotor is working in a normal operating condition with a balanced load, the angular velocities p, q and r have a steady-state operating condition at 0, which has a linearized expression as, for example,

$$qr \approx \bar{q}\bar{r} + \bar{r}(q - \bar{q}) + \bar{q}(r - \bar{r}) = 0$$

, where the steady-state values of $\bar{q} = \bar{r} = 0$. With this assumption, (2.1) is simplified to

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 1/I_{xx} & 0 & 0 \\ 0 & 1/I_{yy} & 0 \\ 0 & 0 & 1/I_{zz} \end{bmatrix} \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} \tag{2.3}$$

Additionally, the steady-state operating conditions for the three Euler angles are chosen to be zero, leading to the linearized dynamics model for (2.2) as

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2.4}$$

In summary, the dynamics models for a quadrotor UAV consist of two sets of integrators in a normal operating condition given by (2.3)-(2.4).

2.1.2 Actuator Dynamics

In quadrotor control, the torques τ_x, τ_y and τ_z in the body frame are generated by the differences in rotor thrusts. The upward thrust produced by each rotor is

$$T_i = b_t \omega_{m_i}^2, \quad i = 1, 2, 3, 4.$$

The total thrust is, hence,

$$T = \sum T_i,$$

where b_t is the thrust constant determined by air density, the length of the blade and the blade radius, ω_{m_i} is the ith rotor's angular speed. Because the altitude of the UAV is not controlled in this paper, the total thrust is manually set by the operator.

The torques about a quadrotor's $x-$axis and $y-$axis are

$$\tau_x = d_{mm}(T_4 - T_2) = d_{mm} b_t (\omega_{m_4}^2 - \omega_{m_2}^2) \tag{2.5}$$

$$\tau_y = d_{mm}(T_3 - T_1) = d_{mm} b_t (\omega_{m_3}^2 - \omega_{m_1}^2), \tag{2.6}$$

where d_{mm} is the distance from the motor to the mass center. The torque applied to each propeller by the motor is opposed by aerodynamic drag and the total reaction torque about the z-axis is

$$\tau_z = k_d(\omega_{m_1}^2 + \omega_{m_3}^2 - \omega_{m_2}^2 - \omega_{m_4}^2), \tag{2.7}$$

where k_d is a drag constant determined by the same factors as b_t.

The relationship between torques, thrust and rotors' angular speed is given in the following matrix form:

$$\begin{bmatrix} \omega_{m_1}^2 \\ \omega_{m_2}^2 \\ \omega_{m_3}^2 \\ \omega_{m_4}^2 \end{bmatrix} = \begin{bmatrix} 1/4b_t & 0 & -1/2d_{mm}b_t & -1/4k_d \\ 1/4b_t & -1/2d_{mm}b_t & 0 & 1/4k_d \\ 1/4b_t & 0 & 1/2d_{mm}b_t & -1/4k_d \\ 1/4b_t & 1/2d_{mm}b_t & 0 & 1/4k_d \end{bmatrix} \begin{bmatrix} T \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} \tag{2.8}$$

From (2.8), once the manipulated variables T, τ_x, τ_y,τ_z are decided by the cascade feedback controllers, the velocities of motors will be uniquely determined because the matrix that relates the manipulated variables to the rotor velocities is invertible. The rotors acting as the actuators in this UAV control application will implement the control actions determined by T, τ_x, τ_y, τ_z, through their reference signals ω_{m_1}, ω_{m_2} ω_{m_3} and ω_{m_4}. The actuator dynamics will also affect the closed-loop control performance, which should be included in the quadrotor model and they are approximated by a first-order transfer function with time delay:

$$\frac{\Omega_{m_i}(s)}{V_i(s)} = \frac{r_{wv}e^{-d_m s}}{\epsilon_m s + 1}, \tag{2.9}$$

where $V_i(s)$ is the Laplace transform of the armature voltage to the ith motor, ϵ_m is the time constant, d_m is the time delay, and r_{wv} is the steady-state gain, which indicates the ratio between the motor's steady-state speed and the motor armature voltage. The armature voltage v_i is changed by manipulating the duty cycle of the PWM signal of each motor drive, where the relationship between the motor armature voltage and the PWM duty cycle is

$$v_i = d_i V_{bat}, \tag{2.10}$$

where d_i is the PWM signal duty cycle of the ith DC motor drive and V_{bat} is the battery voltage assumed to be constant. Substituting equation (2.10) to equation (2.9) yields:

$$\frac{\Omega_{m_i}(s)}{D_i(s)} = \frac{V_{bat} r_{wv} e^{-d_m s}}{\epsilon_m s + 1}, \tag{2.11}$$

which describes the actuator dynamics. Figure 2.2 illustrates the actuator dynamics in a quadrotor control problem. . It should be noted that, the above discussion assumes

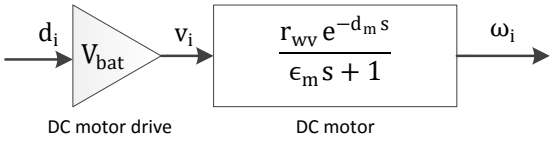


Figure 2.2: Quadrotor actuator dynamics

the the DC motors being the actuators. If other types of motors are adopted, the $V_{bat} r_{wv}$

therm in (2.11) should be replaced by the corresponding motor constants.

2.1.3 Quadrotor Translational Dynamics

Now we define a frame $\{V\}$, which is attached to the vehicle and with the same orientation as $\{B\}$ but with its x- and y- axes parallel to the ground. As $\{V\}$ and $\{B\}$ have the same orientation, the Euler angle ψ from the frame $\{B\}$ to the frame $\{V\}$ is zero. We also assume the frame $\{V\}$ is an inertial frame with zero angular velocity.

The translational dynamics of the quadcopter in the frame $\{V\}$ is given by Newton's second law:

$$m \begin{bmatrix} \ddot{x}^V \\ \ddot{y}^V \\ \ddot{z}^V \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + \boldsymbol{R}_{B\to V} \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix}, \tag{2.12}$$

where $\boldsymbol{R}_{B\to V}$ is the transformation matrix from $\{B\}$ to $\{V\}$ described by the following equation:

$$\boldsymbol{R}_{B\to V} = \begin{bmatrix} C_\psi C_\theta & C_\psi S_\theta S_\phi - S_\psi C_\phi & C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi C_\theta & S_\psi S_\theta S_\phi + C_\psi C_\phi & S_\psi S_\theta S_\phi - C_\psi S_\phi \\ -S_\theta & C_\theta S_\phi & C_\theta C_\phi \end{bmatrix},$$

where $S_x = sin(x),\ C_x = cos(x)$ and $T_x = tan(x)$.

When $\psi = 0$, $\cos\psi = 1$ and $\sin\psi = 0$. $\boldsymbol{R}_{B\to V}$ becomes

$$\boldsymbol{R}_{B\to V} = \begin{bmatrix} C_\theta & S_\theta S_\phi & S_\theta \\ 0 & C_\phi & -S_\phi \\ -S_\theta & C_\theta S_\phi & C_\theta C_\phi \end{bmatrix},$$

and (2.12) becomes

$$\begin{bmatrix} \ddot{x}^V \\ \ddot{y}^V \\ \ddot{z}^V \end{bmatrix} = \frac{1}{m} \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \begin{bmatrix} T\sin\theta \\ -T\sin\phi \\ T \end{bmatrix}. \tag{2.13}$$

When the quadrotor is working in a normal operating condition with a balanced load, Euler angles have a steady-state operating condition at 0, and the thrust has a

steady-state operating condition at mg. With these assumptions, (2.13) is simplified to

$$\begin{bmatrix} \ddot{x}^V \\ \ddot{y}^V \\ \ddot{z}^V \end{bmatrix} = \frac{1}{m} \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \begin{bmatrix} g & 0 & 0 \\ 0 & -g & 0 \\ 0 & 0 & \frac{1}{m} \end{bmatrix} \begin{bmatrix} \theta \\ \phi \\ T \end{bmatrix}. \tag{2.14}$$

2.2 Quadrotor Hardware

In this research, two quadrotor platforms are designed and built from scratch, which are named as quadrotor TC1.0 (Figure 2.3) and TC3.0 (2.4). These two quadrotor platforms share similar frame and flight controller design. The block diagram to illustrate the overall design structure of the quadrotors is shown in Figure 2.5. The quadrotors consist of five main components: RC transmitter/receiver, IMU sensor board, data logger, microprocessor and actuators. The RC transmitter/receiver is to send and receive reference signals. The IMU sensor board is to measure the Euler angles and angular velocities. The data logger is to record flight data such as Euler angles and reference signals. The processor is to generate control signals to stabilize the UAV's attitude. Actuators are to generate thrust and torques, which consist of motor drives, motors, gearboxes and blades.

The main difference between these two platforms is: brushed DC motors are used on TCV1.0 and brushless DC motor are used on TCV3.0. However the speed of both types of motors are controlled by PWM signals. As a result, the dynamic equations derived in the previous section apply on both platforms.

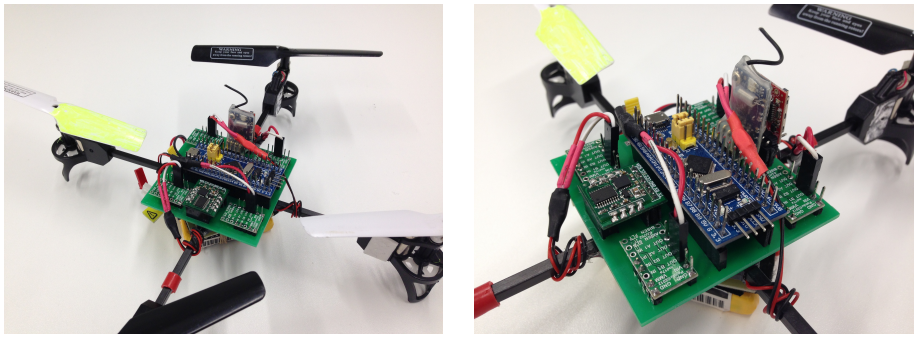


Figure 2.3: Quadrotor TC1.0

Figure 2.4: Quadrotor TC3.0

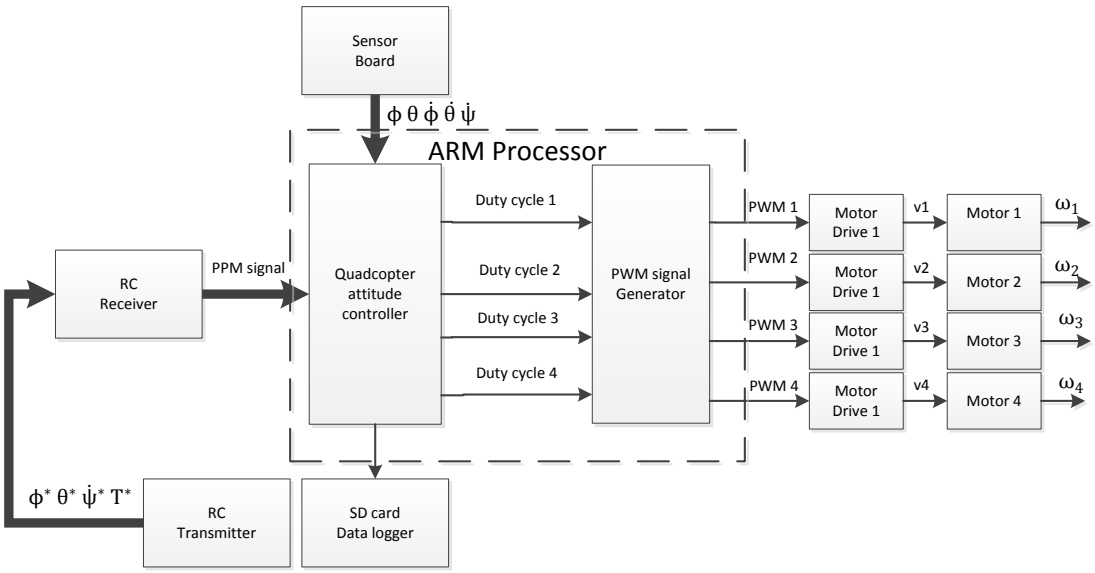


Figure 2.5: Quadrotor hardware design structure

2.2.1 Quadrotor TC1.0 Design

The frame includes four arms in a cross arrangement, with four motors mounted at the end of each arm. The front and back blades are spinning in counter-clockwise and the left and right blades are spinning in clockwise. The flight controller board is placed in the center of the frame to make it symmetric. The Li-Po battery is attached on the bottom of the frame. The microprocessor is STM32F103CB, which is a 72 MHz ARM Cortex-M3 MCU with 128 kilobytes flash. The microprocessor is used to run the control software and most of the sensors and actuator electronics are directly connected

to it via RS-232 or digital IOs. The microprocessor is connected to RC receiver via 4 digital IO ports. The data received from RC receiver is pulse-position modulation (PPM) signal, and the timer 3 in the microprocessor is used to decode the PPM signal. The sensor board and the SD data logger are connected to the microprocessor vis RS 232 serial port. The PWM signals are generated by the microprocessor's timer 2 and connected to four DC motor drives via four digital IO ports.

The sensor board consists of a 6-DOF Inertial measurement unit (IMU) and a 8-bit MCU. The IMU is MPU6505, which has a 3-axis accelerometer and a 3-axis gyroscope. As the accelerometer can only measure angular speed, in order to get the accurate roll and pitch angles, the translational acceleration is needed to perform data fusion. The data fusion is performed on the 8-bit MCU. It should be pointed out that the accurate yaw angle can not be achieved due to the lack of digital compass. As a result, only the angular velocity in $z-$axis can be controlled accurately.

The motor drives are DRV8833 Dual Motor Driver Carrier. Normally, each DRV8833 can be used for bidirectional control of two DC motors at $2.7V$ to $10.8V$. However, it can only supply up to $1.2A$ per channel continuously and tolerate peak currents $2A$, which is lower than what is needed in this application. As a result, the two channels in each DRV8833 are paralleled to deliver $2.4A$ continuous ($4A$ peak) to one motor.

The motors on TC1.0 are 820 coreless DC motors. The motor diameter is $8mm$ and the length is $20mm$. Compared with brushless motors, coreless DC motors and their drives are simpler and lighter. The motor's rated voltage is $5V$.

The SD card data logger is SparkFun OpenLog. It is used to record the flight information such as the reference signals, control signals and output signals. The data logger communicates with the microprocessor via the RS232 serial port and save data to a SD card. The details of the data format recorded by the data logger and the MATLAB script to decode the data is given in Appendix B.

The RC transmitter is WFT06X-A. It has 6 channels and its radio frequency is $2.4Ghz$. Only the first four channels are used to send ϕ^*, θ^*, $\dot{\psi}^*$ and T^* to the RC receiver. Quadrotor TC1.0's components are listed in Table 2.1.

Table 2.1: TC1.0 hardware list

Function	Model
DC motor drive	DRV8833 Dual Motor Driver Carrier
Sensor board	MPU6505
microprocessor	STM32F103C8T6
RC receiver	WFLY065
DC motor	820 Coreless Motor
RC transmitter	WFT06X-A
Data logger	SparkFun OpenLog

2.2.2 Quadrotor TC3.0 Design

Compared with TC1.0, TC3.0 has larger weight and size and it is more robust and more efficient. The frame of TC3.0 has the same rotor arrangement as TC1.0. The major improvement of TC3.0 frame design is that rotors are protected by four plastic rings. Besides, gear boxes are not needed on TC3.0 as brushless DC motors can provide enough torques at high speed. The TC3.0 flight controller board uses the same IMU, micro controller and data logger as TC1.0. Motor drivers are not integrated on the flight controller board.

The motor drives are 12A Afro Electrical Speed Controller(ESC). They can provide maximum $10A$ current to the motors and provide $5V$ regulated power to the flight controller board. The motors on TC3.0 are MultiStar 2206 brushless DC motors. Brushless DC motors provide better frequency than DC motors. Quadrotor TC3.0's components are listed in Table 2.2.

Table 2.2: TC3.0 hardware list

Function	Model
Motor drive	12A Afro ESC
Sensor board	MPU6505
microprocessor	STM32F103C8T6
RC receiver	WFLY065
Brushless DC motor	MultiStar 2206
RC transmitter	WFT06X-A
Data logger	SparkFun OpenLog
Propeller	Diatone Propellers 6040

2.2.3 Quadrotor TC1.0 Physical Parameter Measurement

2.2.3.1 Inertia terms

The method described in [Derafa et al., 2006] and [Armstrong et al., 1986] is used to identify the inertia terms of the quadrotor TC1.0. The experiment setup for identifying the inertia terms is shown in Figure 2.6 and 2.7. As it can be seen, the quadrotor is attached to two wires. The eigenmode of the system is activated by rotating the quadrotor around the axis parallel to the wires. The eigenfrequency at which the system rotates is used to determine the inertia of the quadrotor. According to Derafa et al. [2006] and Armstrong et al. [1986], the inertia term can be calculated using

$$I_{axis} = \frac{mgr_{cm}^2}{\omega_{eigen}^2 l_{wire}}, \tag{2.15}$$

where r_{cm} is the distance from the centre of mass to the wire in meters, l_{wire} is the length of the wire in meters and ω_{eigen} is the eigenfrequency in rad/sec. The results

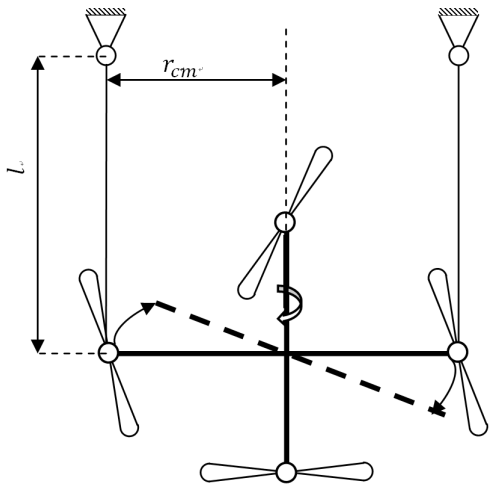


Figure 2.6: Establishing the inertia around the x- and y-axis of TC1.0

of the measurement for the y-axis and z-axis are shown in Figure 2.8. This experiment is performed three times for each axis. The graph clearly shows the oscillation of the quadrotor angular speed, with some degree of damping caused by the air-resistance. However, this resistance is not used in this controller, since it has an insignificant effect on the model.

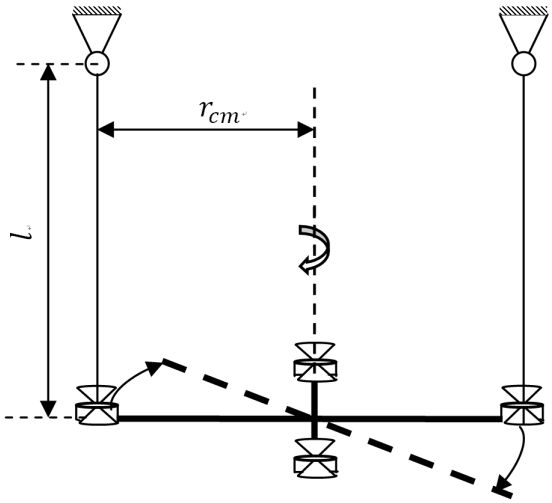
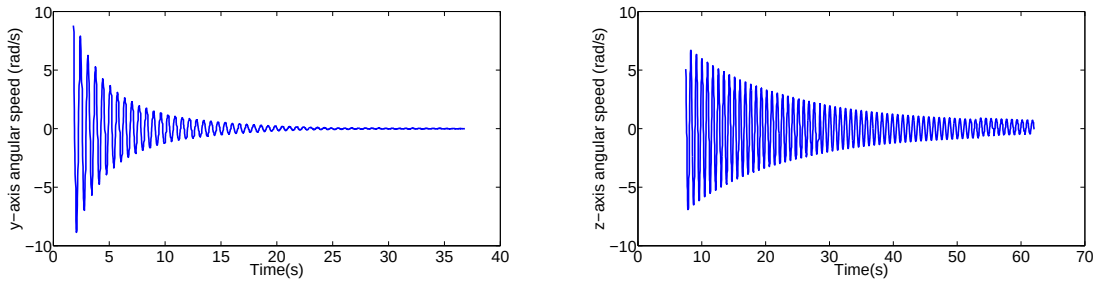


Figure 2.7: Establishing the inertia around the z-axis of TC1.0

To calculate the inertia terms, for example, on y axis, we have the eigenfrequency $\omega_{eigen} = 84.703 rad/s$, $r_{cm} = 0.635m$, $m = 0.145kg$, $d_{mm} = 0.11m$. Substituting all these value to 2.15, we can get $I_{yy} = 3.2e^{-4} kgm^2$.



(a) y-axis (b) z-axis

Figure 2.8: Test results of the eigenfrequencies around y-axis and z-axis

2.2.3.2 Thrust and drag constant

Another experiment is performed to identify the thrust constant b_t and the drag constant k_d. The experiment setup in shown in Figure 2.9. In this experiment, the force generated by one propeller is measured by a scale. The propeller is fixed to a stiff rod. The disc underneath is used to prevent the air generated by the propellers from interfering with the measurements. The relation between the rotational velocity of the blade

and the force generated can be used to determine the blade constant b_t. The results of the measurements are plotted in Figure 2.10.

As we can see from Figure 2.10(a), the thrust is proportional to the squared rotor speed.In Figure 2.10(b), the relationship between the propeller's angular speed ω_m and the motor's armature voltage is almost linear when the voltage is between $2.5V$ and $5.8V$. In quadrotor control, the motor voltage should be limited within the linear region and the ratio between the rotor speed and the motor voltage is assumed to be constant. In Figure 2.10(c), the relation between the thrust and the rotor speed is parabolic.

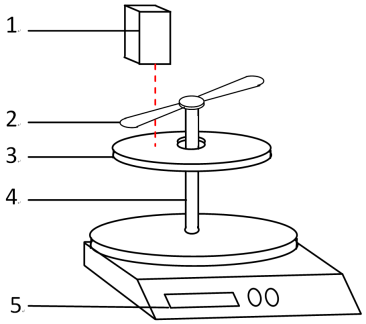


Figure 2.9: Test to determine the thrust constant b. (1) laser tachometer (2) blade (3) disc (4) rod (5) scale

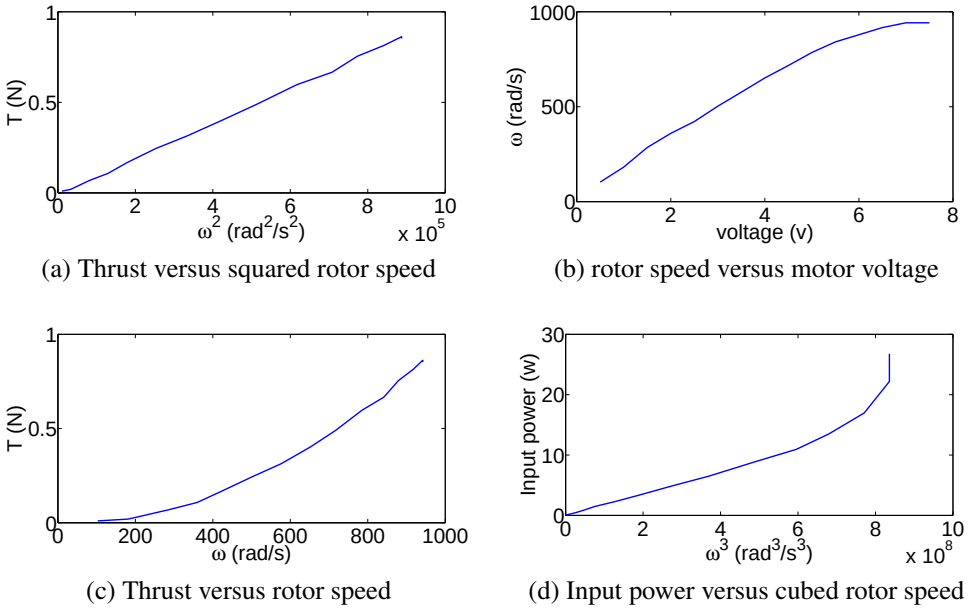
The same experiment set-up can be used for the calculation of the drag coefficient k_d. As the equipment for measuring the torque is not available, the torque is estimated by using the relationship between the motor input power and the propeller's angular speed:

$$\tau = \frac{P_{input}}{\omega_m}, \tag{2.16}$$

where the input power P_{input} is the product of the motor voltage and current. The drag constant k_d is estimated by the following formula:

$$k_d = \frac{P_{input}}{\omega_m^3} \tag{2.17}$$

The test result is shown in Figure 2.10(d) where k_d is estimated from the linear part of the curve.



(a) Thrust versus squared rotor speed

(b) rotor speed versus motor voltage

(c) Thrust versus rotor speed

(d) Input power versus cubed rotor speed

Figure 2.10: Physical parameter measurement experimental results of TC1.0

2.2.3.3 DC motor parameters

In order to measure the DC motor's time time constant ϵ_m, a step response experiment is performed and the result is shown in Figure 2.11. As it can be seen, the time taken to reach 63.2% of its steady-state value is $72ms$. It is also interesting to see that there is a $32ms$ time delay. This time delay should be from the DC motor drive used in experiment.

The physical parameters of TC1.0 are summarised in Table 2.3. As model based controller design is only used on TC1.0, the physical parameters of TC3.0 are not measured in this thesis. However, TC3.0 will be used in the automatic tuning of PID controller parameters.

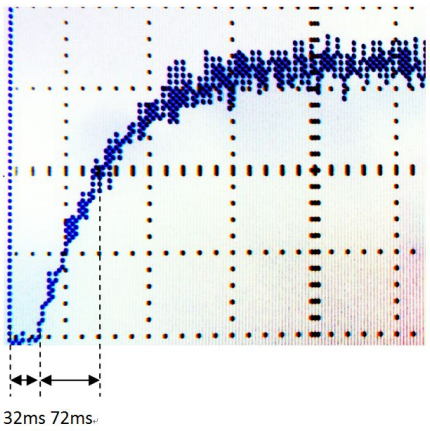


Figure 2.11: DC motor step response

Table 2.3: Quadrotor TC1.0 parameters

parameters	**value**
I_{xx}	$3.2e^{-4}\ kgm^2$
I_{yy}	$3.2e^{-4}\ kgm^2$
I_{zz}	$4.6e^{-4}\ kgm^2$
b_t	$9.7e^{-7}$
k_d	$2.5e^{-9}$
r_{cm}	$0.565m$
m	$0.145\ kg$
d_{mm}	$0.110\ m$
V_{bat}	$8.28\ v$
r_{wv}	$137.6571\ rad/vs$
$\bar{\omega}_m$	$606.2469\ rad/s$
d_s	$0.032\ s$
ϵ_m	$0.072\ s$

2.3 Summary

In this chapter the quadrotor's dynamics equations are derived, which will be utilized to design quadrotor controllers in the following chapters. Quadrotor TC1.0 and TC3.0 hardware designs are also described, and only the physical parameters of TC1.0 are measured. These two quadrotor platforms and TC1.0's physical parameters will be used throughout this thesis for controller performance analysis, comparisons and simulations.

Chapter 3

Model-based Quadrotor PID Controller Design

3.1 Introduction

In this chapter, the conventional model-based quadrotor cascade PI controller design method is proposed. The cascade control structure is deployed to control quadrotor TC1.0. As TC1.0 is only equipped with Euler angle and angular velocity sensors, only the attitude controllers are designed and implemented.

In this chapter, firstly the transfer functions of the quadrotor's three channels are derived. Secondly, both inner-loop and outer-loop controllers are designed based on the simplified integrator model, which is the common practice in the flight controller design. In order to analyze the effects of the neglected dynamics, the closed-loop frequency responses and step responses are used. Lastly, the hardware implementation of the conventional quadrotor closed-loop control system is presented. In the end, flying test results are given to show that the overall performance of the closed-loop system is very poor, leading to the motivation for proposing the innovative automatic tuning of PI cascade control system as in Chapter 4.

3.2 Quadrotor Attitude Controller Design

The general closed-loop quadrotor control system is illustrated in Figure 3.1, where a cascade control structure is used in the control system configuration. The advantages of using a cascade structure[Wang et al., 2015] in quadrotor controller design are discussed below.

1. Simplification of control system design. When using a cascade control system, the complex quadrotor plant is decomposed into a few subsystems. Because of the large difference between the performance requirements of the subsystems, the PI controllers can be designed separately and effectively using the models of the subsystems.

2. Effectively handling nonlinearity of the inner-loop system. As shown in Chapter 2, there are nonlinearities in the angular velocity subsystem. With the inner-loop control, a high gain feedback control is used in practice to overcome the effects of nonlinearities.

3. Reducing computational cost. In the cascade control system design, the inner-loop system will have a much larger bandwidth than the one used in the outer-loop control system. Dual sampling rate can be used in the implementation of the cascade control system in which the outer-loop control system is sampled in a much slower rate. As a result, the computational cost for the entire control system is reduced.

Two PI controllers are deployed to control roll and pitch angles in order to eliminate steady-state errors in the presence of disturbance. It should be noted that in this thesis, the yaw angle is not controlled because there is no yaw angle measurement due to the lack of a digital compass on the quadrotor. Instead, the yaw angular velocity is controlled, as the angular velocity in the $z-$axis can be measured accurately by the gyroscope.

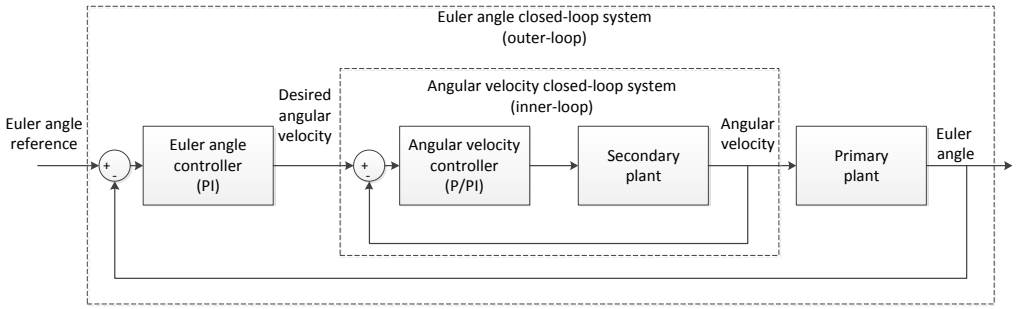


Figure 3.1: Cascade feedback control structure

In order to design the inner-loop feedback controllers, the following linearization of the physical model is examined. Substituting (2.5), (2.6) and (2.7) to (2.3) gives:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 1/I_{xx} & 0 & 0 \\ 0 & 1/I_{yy} & 0 \\ 0 & 0 & 1/I_{zz} \end{bmatrix} \begin{bmatrix} d_{mm} b_t (\omega_{m_4}^2 - \omega_{m_2}^2) \\ d_{mm} b_t (\omega_{m_3}^2 - \omega_{m_1}^2) \\ k_d (\omega_{m_1}^2 + \omega_3^2 - \omega_{m_2}^2 - \omega_{m_4}^2) \end{bmatrix}, \tag{3.1}$$

and from first-order Taylor series expansion, $\omega_{m_i}^2$ can be approximated as follows:

$$\omega_{m_i}^2 \approx 2\bar{\omega}_m \omega_{m_i} - \bar{\omega}_m^2, \tag{3.2}$$

where $\bar{\omega}_m$ is the motor's rated speed. Substituting (3.2) to (3.1) we get:

$$\dot{p} = \frac{2 d_{mm} b_t \bar{\omega}_m}{I_x} (\omega_{m_4} - \omega_{m_2}) \tag{3.3}$$

$$\dot{q} = \frac{2 d_{mm} b_t \bar{\omega}_m}{I_y} (\omega_{m_3} - \omega_{m_1}) \tag{3.4}$$

$$\dot{r} = \frac{2 k_d \bar{\omega}_m}{I_z} (\omega_{m_1} + \omega_{m_3} - \omega_{m_4} - \omega_{m_2}). \tag{3.5}$$

Performing Laplace transform we get

$$sP(s) = \frac{2d_{mm}b_t\bar{\omega}_m}{I_x}(\Omega_4(s) - \Omega_2(s)) \tag{3.6}$$

$$sQ(s) = \frac{2d_{mm}b_t\bar{\omega}_m}{I_y}(\Omega_3(s) - \Omega_1(s)) \tag{3.7}$$

$$sR(s) = \frac{2k_d\bar{\omega}_m}{I_z}(\Omega_1(s) + \Omega_3(s) - \Omega_4(s) - \Omega_2(s)). \tag{3.8}$$

Substituting (2.11) to (3.6)−(3.8) gives

$$sP(s) = \frac{2d_{mm}b_t\bar{\omega}_m}{I_x}\frac{V_{bat}r_{wv}e^{-d_m s}}{\epsilon s + 1}(D_4(s) - D_2(s)) \tag{3.9}$$

$$sQ(s) = \frac{2d_{mm}b_t\bar{\omega}_m}{I_y}\frac{V_{bat}r_{wv}e^{-d_m s}}{\epsilon s + 1}(D_3(s) - D_1(s)) \tag{3.10}$$

$$sR(s) = \frac{2k_d\bar{\omega}_m}{I_z}\frac{V_{bat}r_{wv}e^{-d_m s}}{\epsilon s + 1}(D_1(s) + D_3(s) - D_4(s) - D_2(s)). \tag{3.11}$$

Now the four new desired control signals u_T, u_x, u_y and u_z are defined as

$$\begin{aligned} u_T &= T^* \\ u_x &= d_4 - d_2 \\ u_y &= d_3 - d_1 \\ u_z &= d_1 + d_3 - d_4 - d_2, \end{aligned}$$

and the relationships between these desired control signals and the PWM duty cycles are described by the following equations:

$$\begin{bmatrix} D_1 \\ D_2 \\ D_3 \\ D_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 & -1 \\ 1 & -1 & 0 & 1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_T \\ u_x \\ u_y \\ u_z \end{bmatrix}. \tag{3.12}$$

Finally, the transfer functions of the three angular velocities are given as

$$G_p(s) = \frac{P(s)}{U_x(s)} = \frac{2d_{mm}b_t\bar{\omega}_m}{I_x}\frac{V_{bat}r_{wv}e^{-d_m s}}{\epsilon s^2 + s} \tag{3.13}$$

$$G_q(s) = \frac{Q(s)}{U_y(s)} = \frac{2d_{mm}b_t\bar{\omega}_m}{I_y}\frac{V_{bat}r_{wv}e^{-d_m s}}{\epsilon s^2 + s} \tag{3.14}$$

$$G_r(s) = \frac{R(s)}{U_z(s)} = \frac{2k_d\bar{\omega}_m}{I_z}\frac{V_{bat}r_{wv}e^{-d_m s}}{\epsilon s^2 + s}. \tag{3.15}$$

Three controllers can be designed independently for each axis based on the above transfer functions. As all above transfer functions are in the same form, they can be represented by a single transfer function, which is used for the design of inner-loop controller, as

$$G_{in}(s) = \frac{b_0 e^{-d_m s}}{a_2 s^2 + a_1 s}. \tag{3.16}$$

If the motor's dynamics and time delay are ignored, the transfer function of the inner-loop plant is:

$$\bar{G}_{in}(s) = \frac{b_0}{a_1 s}. \tag{3.17}$$

For the outer-loop system, transfer functions of ϕ and θ are given as

$$G_\phi(s) = \frac{1}{s}G_p(s) \tag{3.18}$$

$$G_\theta(s) = \frac{1}{s}G_q(s). \tag{3.19}$$

If the inner-loop controlled dynamics are much faster than the outer-loop dynamics, the inner-loop dynamics can be neglected and the outer open-loop transfer function is written as:

$$\bar{G}_{out}(s) = \frac{1}{s}. \tag{3.20}$$

In summary, both the inner and the outer open-loop dynamics can be approximated by a integrator with a DC gain K:

$$G(s) = \frac{K}{s}. \tag{3.21}$$

Now both the inner-loop and outer-loop controllers can be designed by using (3.21).A PI controller's transfer function is given by

$$C(s) = K_c(1 + \frac{1}{\tau_I s}), \tag{3.22}$$

which can be written in the transfer function form,

$$C(s) = \frac{K_p s + K_i}{s}, \tag{3.23}$$

where $K_p = K_c$, $K_i = \frac{K_c}{\tau_I}$. A model-based PI controller design method is used to calculate K_p and K_i [Liuping Wang, 2015] for both inner-loop and outer-loop systems. The solution of the PI controller parameters is calculated by equating the desired closed-loop poles to the actual closed-loop poles. The locations of the closed-loop poles determine whether the closed-loop system is stable, the closed-loop response time and the bandlimit.

The actual closed-loop system is calculated as

$$T_{cl} = \frac{G(s)C(s)}{1 + G(s)C(s)} = \frac{\frac{K}{s}\frac{K_p s + K_i}{s}}{1 + \frac{K}{s}\frac{K_p s + K_i}{s}} = \frac{K(K_p + K_i)}{s^2 + K(K_p s + K_i)}. \tag{3.24}$$

The closed-loop poles of the actual system are the solutions of the polynomial equation with respect to s

$$s^2 + b(K_p s + K_i) = 0. \tag{3.25}$$

To find the controller parameters K_p and K_i, the following polynomial equation is set,

$$s^2 + b(K_p s + K_i) = s^2 + 2\zeta\omega_n s + \omega_n^2, \tag{3.26}$$

where the left-hand side of the equation is the polynomial that determines the actual closed-loop poles and the right-hand side is the polynomial that determines the desired closed-loop poles. ζ is the desired damping ratio and ω_n is the desired natural frequency. Typically, ζ is selected as either 0.707 or 1. By equating these two polynomials, the actual closed-loop poles are assigned to the desired closed-loop poles. Solving (3.26) gives

$$K_p = \frac{2\zeta\omega_n}{K} \tag{3.27}$$

$$K_i = \frac{\omega_n^2}{K}. \tag{3.28}$$

When tuning the inner-loop system, the actuator's stable pole was ignored as the integrator in (3.16) has much slower response than the one from the stable pole. Furthermore, the time delay is ignored if the desired closed-loop dynamics is much longer than the time delay. So the transfer function (3.17) is used to design the inner-loop controller. The closed-loop transfer function for the inner-loop system is given as:

$$\bar{T}_{in}(s) = \frac{\bar{G}_{in}(s)C_{in}(s)}{1 + \bar{G}_{in}(s)C_{in}(s)} \tag{3.29}$$

, where $C_{in}(s)$ is the inner-loop controller transfer function. However, if the motor's dynamics and time delay is not ignored, the closed-loop transfer function for the inner-loop system is given as:

$$T_{in}(s) = \frac{G_{in}(s)C_{in}(s)}{1 + G_{in}(s)C_{in}(s)}. \tag{3.30}$$

Similarly, when designing the outer-loop controller, the inner-loop dynamics is often ignored. The closed-loop transfer function for the outer-loop system is given as:

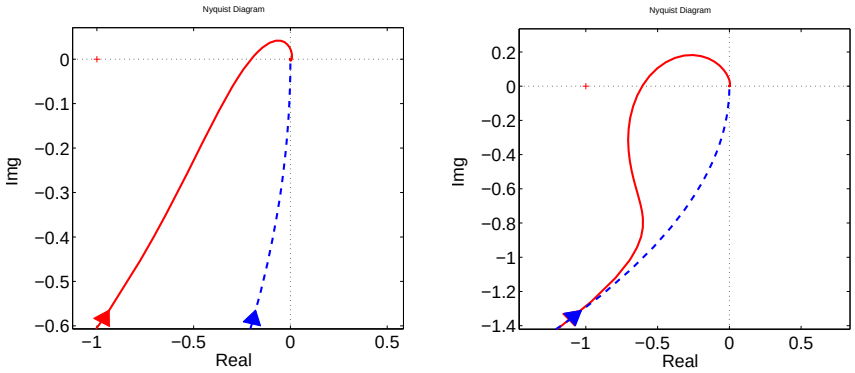
$$\bar{T}_{out}(s) = \frac{\bar{G}_{out}(s)C_{out}(s)}{1 + \bar{G}_{out}(s)C_{out}(s)}, \tag{3.31}$$

where $C_{out}(s)$ is the outer-loop controller transfer function.

However, if the inner-loop dynamics is considered when designing the outer-loop controller, the closed-loop transfer function for the outer-loop system is given as:

$$T_{out}(s) = \frac{\bar{G}_{out}(s)T_{in}(s)C_{out}(s)}{1 + \bar{G}_{out}(s)T_{in}(s)C_{out}(s)} \tag{3.32}$$

In order to analyze whether the neglected dynamics has an effect on the closed-loop systems, the open-loop frequency responses from the TC1.0's inner-loop and outer-loop systems are computed. The values of the quadrotor's physical parameters are given in Table 2.3. The Nyquist diagram of the inner-loop system is shown in Figure 3.2(a). The solid line is the frequency response of the actual inner open-loop system; the dotted line is the simplified inner-loop system.



(a) inner-loop (b) outer-loop

Figure 3.2: Quadrotor open-loop frequency response comparison.Key: solid lines actual frequency response; dotted lines frequency response with ignored dynamics

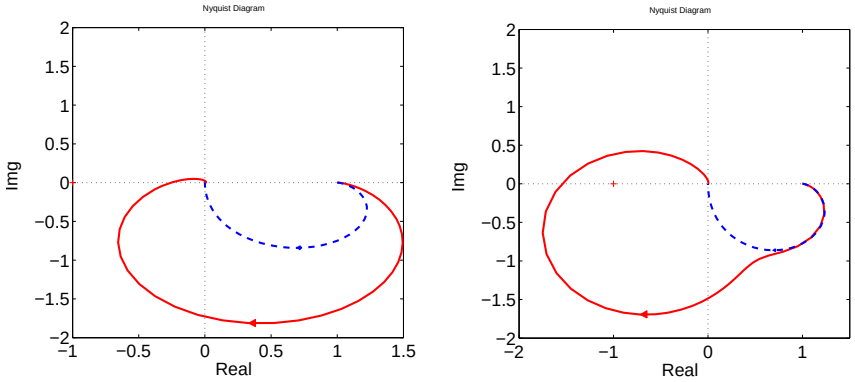
We can make several observations from Figure 3.2. First of all, as all the Nyquist curves do not enclose -1 on the real axis, all systems are stable. However, as the actual systems' Nyquist curves (solid lines in Figure 3.2) are quite different from the simplified systems' Nyquist curves (dotted lines in Figure 3.2), the actual and simplified closed-loop systems are expected to exhibit different performance.

In order to further analyze the effects of the neglected dynamics on the closed-loop systems' performance, the closed-loop frequency responses from the quadrotor TC1.0's inner-loop and outer-loop systems are computed. The frequency response of the inner-loop system is shown in Figure 3.3(a). The solid line is the frequency response of the actual inner-loop system described by (3.30); the dotted line is the simplified inner-loop system described by (3.29).

The frequency response of the outer-loop system is shown in Figure 3.3(b). The solid line is the frequency response of the actual outer closed-loop system described by (3.32); the dotted line is the simplified outer closed-loop system described by (3.31).

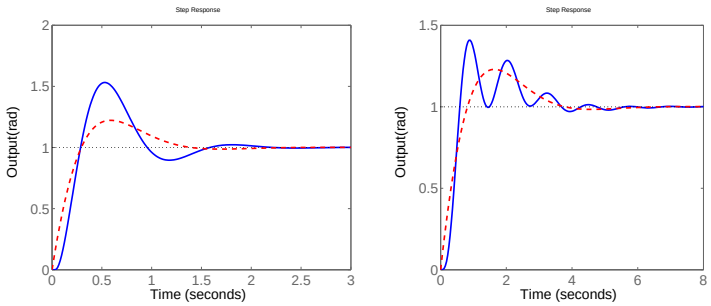
As we can see, the frequency responses of inner and outer closed-loop systems, with and without the actuator dynamics, are very different, which implies the ignored dynamics has an large effect on the quadrotor closed-loop systems, and thus has to be considered in the controller design.

Furthermore, the inner-loop and outer-loop step responses are given in Figure 3.4, which also show obvious difference on the actual system and the approximated system



(a) inner-loop (b) outer-loop

Figure 3.3: Quadrotor TC1.0 frequency response comparison.Key: solid lines actual frequency response; dotted lines frequency response with ignored dynamics



(a) inner-loop (b) outer-loop

Figure 3.4: Quadrotor TC1.0 closed-loop step response comparison.Key: solid lines actual step response; dotted lines step response with ignored dynamics

with neglected dynamics.

3.3 Flight Test

3.3.1 Hardware Implementation of Quadrotor Closed-loop Control

The PI controller parameters calculated by using the model-based design method are tested on the quadrotor V1.0. This quadrotor platform has 3-axis accelerometers and

3-axis gyroscopes, so it is capable of tracking roll and pitch angle references and yaw angular rate reference. The closed-loop control block diagram is given in Figure 3.5. The roll and pitch PI controllers compare the difference between the reference signals ϕ^*, θ^* and the feedback signals ϕ, θ and then generate the angular velocity reference signals $p*$, q^*. The three angular velocity controllers then compare the reference signals and the feedback signals to generate the control signals $u_{1,2,3,4}$. Then the control signals are converted to duty cycles of the PWM generator. The PWM generator generates PWM signals according to the duty cycles and send them to the motor drivers. The motor drivers supply variable voltages to the DC motors to control their speeds.

The flowchart of the control firmware is given in Figure 3.6. Firstly, the system initialization is performed to set all hardware components ready. Then the system waits until the thrust command becomes smaller than 10% of the maximum thrust to start the control loop. Then the reference signals ϕ^*, θ^* and $\dot{\psi}^*$ are manually changed using a RC transmitter. These reference signals are then received by the RC receiver on the quadrotor and sent to the on-board microprocessor. The microprocessor compares the difference between the reference signals from the receiver and the actually feedback signals from the sensors and generate the manipulated variables to stabilize the quadrotor system. The the quadrotor's data logger records the flight test data, including reference signals, feedback signals, manipulated variables, sampling instance and the controller parameters. Here 'itSign' is to determine the sampling interval. Every time 'itSign' becomes positive the inner-loop controller runs once. 'OUTER_ LOOP_ FREQUENCY_ DIVIDER' is to set how many times the outer-loop is slower than the inner-loop. As TC1.0 and TC3.0 share the same control firmware, this flowchart also applies to TC3.0. The C code of PID controllers is given in Appendix D.

3.3.2 Flight test results

The quadrotor's PI controller parameters were initially calculated based on the physical parameters given in Table 2.3, and the simplified integrator model. The resultant initial PI parameters are listed in Table 3.1. However, those PI parameters did not offer good closed-loop control performance, because of the ignored dynamics and the inaccurate physical parameters measured.

Those initial PI parameters had to be adjusted by trial and error and the final con-

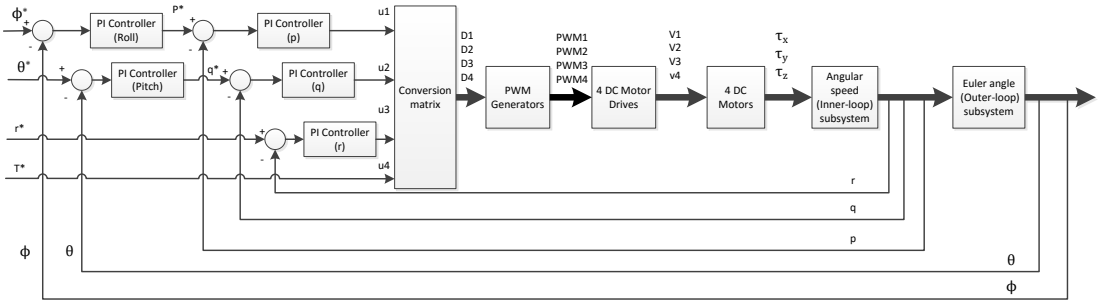


Figure 3.5: Quadrotor closed-loop control system block diagram

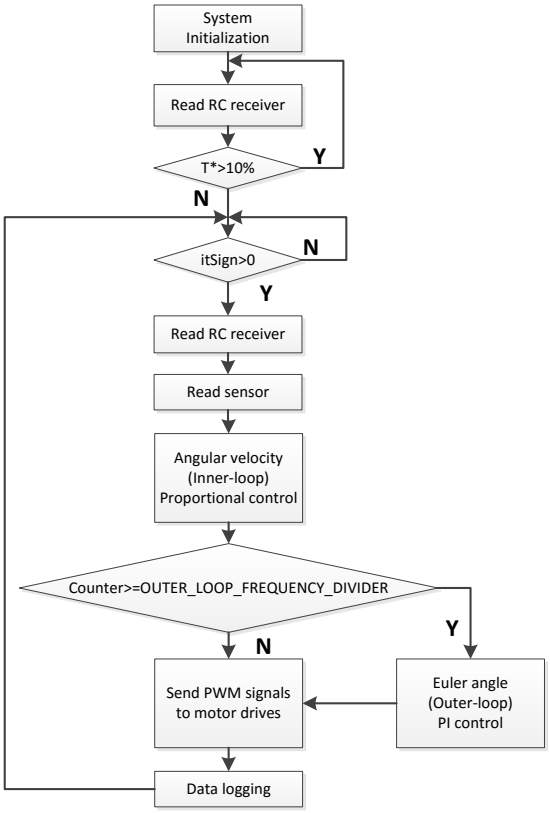


Figure 3.6: Quadrotor control software flowchart

troller parameters are listed in Table 3.2. Those PI parameters were evaluated in a flight test. As it can be seen in Figure 3.7 and 3.8, both Euler angles and angular velocities can follow the reference signals, but errors exist.

The above controller tuning process shows a common difficulty in the conventional

Table 3.1: TC1.0 PI controller initial parameters

Inner Loop
$\omega_n = 15\ \zeta = 1$
Roll rate $K_p = 0.0656\ K_i = 0.4917$
Pitch rate $K_p = 0.0656\ K_i = 0.4917$
Yaw rate $K_p = 0.0437\ K_i = 0.2185$
Outer Loop
$\omega_n = 1.5\ \zeta = 1$
Roll $K_p = 3\ K_i = 2.25$
Pitch $K_p = 3\ K_i = 2.25$

Table 3.2: TC1.0 PID controller parameters tuned by trial- and-error

Inner Loop
$\omega_n = 10\ \zeta = 1$
Roll rate $K_p = 0.0437\ K_i = 0.2185$
Pitch rate $K_p = 0.0437\ K_i = 0.2185$
Yaw rate $K_p = 0.0437\ K_i = 0.2185$
Outer Loop
$\omega_n = 1\ \zeta = 1$
Roll $K_p = 2\ K_i = 1$
Pitch $K_p = 2\ K_i = 1$

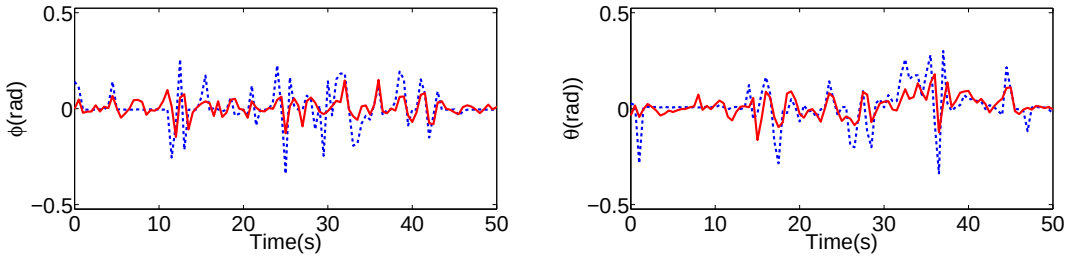


Figure 3.7: Quadrotor TC1.0 Euler angles control with PI controllers. Key: dotted line-reference; solid line- measured output

quadrotor controller design. When using the simplified integrator model, the initial PI controller parameters calculated by the model-based design method can hardly fulfil the control performance requirements. Those initial parameters have to be carefully adjusted with human intervention. The tuning process is quite time-consuming and the tuning results largely depend on the control engineer's experience.

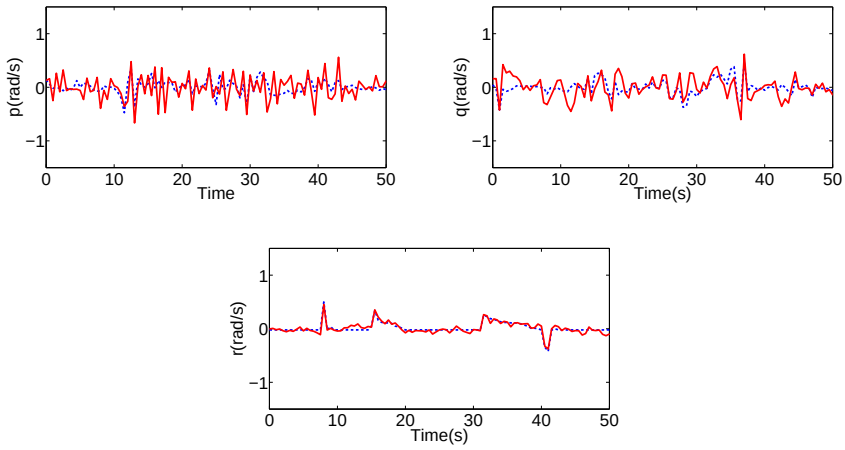


Figure 3.8: Quadrotor TC1.0 Euler angular velocities PI controllers. Key: dotted line- reference; solid line- measured output

3.4 Summary

In this chapter, the conventional model-based controller design of the quadrotor is examined. The quadrotor's primary and secondary plant models with actuator dynamics are derived. In the conventional way, the simplified integrator model is used for both quadrotor inner-loop and outer-loop control design. For the inner-loop system, the PI controllers are designed based on the integrator model by ignoring the actuator dynamics. For the outer-loop system, the PI controllers are design based on the integrator model by ignoring the inner-loop dynamics. Both open-loop and closed-loop frequency response analysis is performed on the inner-loop and outer-loop systems to determine the effects of the ignored dynamics. The results show that the ignored dynamics dramatically change the closed-loop frequency responses of the systems. In other words, the actual closed-loop dynamics can significantly differ from the expected ones, when the actuator or inner-loop dynamics are ignored. Step response simulations performed on the outer-loop also confirm this conclusion. Even though the inner-loop and outer-loop controller could be designed by taking the actuator or inner-loop dynamics into account, one would have to deal with higher order system by using some kind of model order reduction, which could complicate the design process. More importantly, even if we perform model-based design and consider the dynamics of all physical components, the inaccurate physical parameters (i.e. the model mismatch-

ing problem) may still seriously degrade the controller performance. In the quadrotor TC1.0 flying test, the initial PI controller parameters calculated based on the integrator model and the the measured physical parameters failed to meet the control performance requirements. In summary, the conventional model-based quadrotor flight controller design cannot guarantee a good closed-loop control performance. This chapter provides the motivation to seek for a better PID tuning method. In the next chapter, a novel PID controller design method will be introduced to solve these problems.

Chapter 4

Quadrotor PID Controller Automatic Tuning

4.1 Introduction

In the previous chapter, a cascade PI control structure was proposed on the quadrotor TC1.0 platform to control its attitude. Model-based controller design was used to calculate PI controller parameters. However, as it has been shown that the controller design based on the simplified integrator model may not offer desired closed-loop control performance. In addition to that, using a high order transfer function increases the controller design complexity and the model mismatching problem may also degrade the closed-loop control performance.

In this chapter, a simple and effective PID controller auto-tuning process is developed to solve these problems. This chapter is organized as follows. Section 4.2 describes the four steps in the auto-tuner design: relay feedback control, recursive estimation of frequency response, estimation of integrating plus delay model and PID controller design using the tuning rules. In section 4.3, two groups of numerical simulations are conducted to validate the proposed auto-tuning method. In section 4.5, the proposed auto-tuner is applied on quadrotor TC1.0 and TC3.0 platforms to tune their inner-loop and outer-loop controllers. The gain margin and phase margin of quadrotor PD, PI and PID controllers are compared, by using the Nyquist plot. TC3.0's outer-loop step responses when using different PID controller structures are compared in

term of their mean squared errors (MSE). Flight test results are also given in this section to further validate the proposed auto-tuning method. Section 4.7 concludes this chapter.

4.2 Design of Auto-tuner for PID Controllers

This section presents the design of auto-tuner for PID control of integrating systems. The auto-tuner consists of four components: relay feedback control, recursive estimation of the frequency response, estimation of the integrating plus delay model and PID controller design. The first three steps are to fit the system's dynamics into an integrator plus delay model (i.e. model identification stage). The last step is to calculate PID parameters based on the identified model (i.e. controller design stage). When the auto-tuner is applied to tune a cascade control system, the algorithm remains the same for both inner-loop and outer-loop controllers.

4.2.1 Relay Feedback Control

Unlike many other physical systems, quadrotors are open-loop unstable systems, thus, the dynamic response test must be under closed-loop control with an appropriate input excitation signal [Garnier and Wang, 2008; Ljung, 2007; Wang et al., 2005, 2002; Young, 2011; Young et al., 2009]. A proportional controller with known gain K_p is used to stabilize the integrating system and a relay feedback control system is deployed for the output of the closed-loop system. Figure 4.1 shows the configuration of the closed-loop system under relay feedback control. The reference signal $r(t)$ is a constant that represents the steady-state operation of the plant. ϵ is the hysteresis selected to avoid the possible random switches caused by the measurement noise and a is the amplitude of the relay. The signal $\bar{y}(t)$ represents the actual output measurement. The closed-loop control is at the steady-state operation before the relay feedback control is switched on. To ensure a bumpless transfer to the relay feedback control, the same reference signal $r(t)$ is used as the reference signal for the relay-feedback control. Assuming that at the sampling time t_0 the relay feedback control is enabled. The sampling interval is defined as $\Delta t = t_k - t_{k-1}$. To initialize the relay element, at the sampling time t_{-1}, $\bar{u}(t_{-1}) = r(t_{-1})$. For all $t_k \geq t_0$, the calculation of the reference

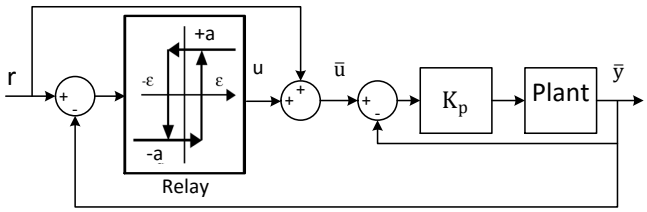


Figure 4.1: Block diagram of relay feedback control.

signal to the closed-loop control system uses the following relay switching rules:

1. Calculate the relay feedback error: $e(t_k) = r(t_k) - \bar{y}(t_k)$.
2. If $|e(t_k)| \leq \epsilon$; then $\bar{u}(t_k) = \bar{u}(t_{k-1})$.
3. If $|e(t_k)| > \epsilon$; then $\bar{u}(t_k) = r(t_k) + a \times sign(e(t_k))$.

This relay feedback control system will produce a sustained oscillation illustrated by the experimental data in Figure 4.1 [Åström and Hägglund, 1984; Atherton, 1975]. Assume that the period of the oscillation is T_p. It is well known that the frequency of the periodic signal $\bar{u}(t)$, denoting by $\omega_1 = \frac{2\pi}{T_p}$, approximately corresponds to the frequency illustrated on the Nyquist curve shown in Figure 4.2, which approximately has $-\frac{\pi\epsilon}{4a}$ as the imaginary part.

4.2.2 Recursive Estimation of Frequency Response

To estimate the closed-loop frequency response

$$T(j\omega_1) = \frac{K_p G(j\omega_1)}{1 + K_p G(j\omega_1)}$$

where $G(j\omega_1)$ is the open-loop frequency response at ω_1, the pair of input and output signals corresponding to the relay feedback control system is used. That is, the input signal equals the relay output signal:

$$u(t) = \bar{u}(t) - r(t) = a \times sign(e(t))$$

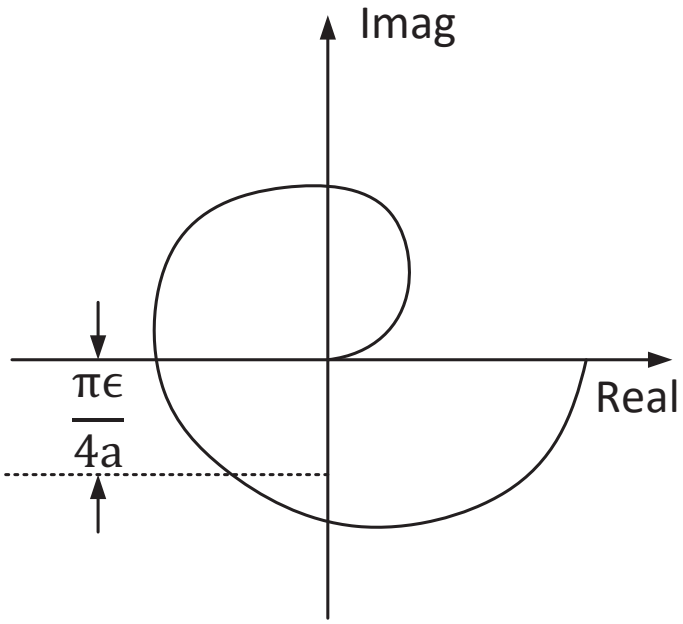


Figure 4.2: Location of $\omega_1 = \frac{2\pi}{T_p}$ on a Nyquist curve.

and the closed-loop output signal with steady-state removed becomes

$$y(t) = \bar{y}(t) - r(t) = -e(t)$$

For a period T_p, the Fourier series expansion of the periodic input signal $u(t)$, is expressed as [Kreyszig and Norminton, 2014]

$$u(t) = \frac{4a}{\pi}(\sin\frac{2\pi}{T_p}t + \frac{1}{3}\sin\frac{6\pi}{T_p}t + \frac{1}{5}\sin\frac{10\pi}{T_p}t + \ldots) \tag{4.1}$$

By choosing sampling interval Δt and the number of samples within one period $N = \frac{T_p}{\Delta t}$, the discretized input signal $u(t)$ at sampling instant $t_k = k\Delta t$ becomes

$$u(k) = \frac{4a}{\pi}(\sin\frac{2\pi k}{N} + \frac{1}{3}\sin\frac{6\pi k}{N} + \frac{1}{5}\sin\frac{10\pi k}{N} + \ldots) \tag{4.2}$$

Noting that the sinusoids can also be expressed in exponential form as

$$\sin\frac{2\pi k}{N} = \frac{1}{2j}(e^{j\frac{2\pi k}{N}} - e^{-j\frac{2\pi k}{N}})$$

equation (4.2) has an alternative expression as

$$\begin{aligned} u(k) &= \frac{2a}{j\pi}\left((e^{j\frac{2\pi k}{N}} - e^{-j\frac{2\pi k}{N}}) + \frac{1}{3}(e^{j\frac{6\pi k}{N}} - e^{-j\frac{6\pi k}{N}}) + \frac{1}{5}(e^{j\frac{10\pi k}{N}} - e^{-j\frac{10\pi k}{N}}) + \dots\right) \\ &= \frac{2a}{j\pi}(e^{j\omega_d k} + \frac{1}{3}e^{j3\omega_d k} + \frac{1}{5}e^{j5\omega_d k} + \dots \\ &- (e^{-j\omega_d k} + \frac{1}{3}e^{-j3\omega_d k} + \frac{1}{5}e^{-j5\omega_d k} + \dots)) \end{aligned} \tag{4.3}$$

where for notational simplicity, $\omega_d = \frac{2\pi}{N}$ denoting the fundamental frequency in discrete-time.

For a stable system with transfer function $T(z)$, in general, it has the z-transfer function model in frequency sampling filter form:

$$T(z) = \sum_{l=-\frac{N-1}{2}}^{\frac{N-1}{2}} T(e^{jl\omega_d})F^l(z), \tag{4.4}$$

where $F^l(z)$ is the lth frequency sampling filter given by

$$\begin{aligned} F^l(z) &= \frac{1}{N}\frac{1-z^{-N}}{1-e^{jl\omega_d}z^{-1}} \\ &= \frac{1}{N}(1 + e^{jl\omega_d}z^{-1} + \dots + e^{j(N-1)l\omega_d}z^{-(N-1)}). \end{aligned}$$

The output of the closed-loop control system under relay feedback with input $u(k)$ is expressed as

$$y(k) = \sum_{l=-\frac{N-1}{2}}^{\frac{N-1}{2}} T(e^{jl\omega_d})f^l(k) + v(k) \tag{4.5}$$

where $f^l(k)$ is the output of the lth frequency sampling filter (Wang and Cluett [2000]) and $v(k)$ is the output measurement noise assumed to be Gaussian distributed with zero mean and variance σ^2.

If the sampling filters are assumed to have zero initial conditions then the outputs of the frequency sampling filters in response to the input signal $u(k)$ satisfy the following

relationships:

$$f^l(k) = \begin{cases} 0, & \text{if } l = 0, \pm 2, \pm 4, \pm 6, \pm 8, \dots \\ \frac{2a}{j\pi|l|} e^{jl\omega_d k}, & \text{if } l = \pm 1, \pm 3, \pm 5, \pm 7, \pm 9, \dots \end{cases} \tag{4.6}$$

This can be verified by calculating the output of the lth filter to the input signal $e^{jl\omega_d k}$ at sampling time k as

$$\begin{aligned} f^l(k) &= \frac{1}{N}(1 + e^{jl\omega_d} q^{-1} + \dots + e^{j(N-1)l\omega_d} q^{-(N-1)}) \frac{2a}{j\pi|l|} e^{jl\omega_d k} \\ &= \frac{2a}{j\pi|l|} e^{jl\omega_d k} \end{aligned} \tag{4.7}$$

where q^{-1} is the backward shift operator. However, the outputs of the rest of the filters are zero in response to the input signal $e^{jl\omega_d k}$ using a similar calculation to the derivation in (4.7).

Since the frequency sampling filters with even numbers are zero in response to the relay feedback control signal $u(k)$, they should be removed from the sum in (4.5), leading to

$$\begin{aligned} y(k) &= T(e^{j\omega_d}) f^1(k) + T(e^{-j\omega_d}) f^{-1}(k) + T(e^{j3\omega_d}) f^3(k) + T(e^{-j3\omega_d}) f^{-3}(k) \\ &+ T(e^{j5\omega_d}) f^5(k) + T(e^{-j5\omega_d}) f^{-5}(k) + \dots + v(k) \end{aligned} \tag{4.8}$$

Because the magnitude of the filter output is inversely proportion to the parameter $|l|$ and additionally $|T(e^{jl\omega_d}|$ decreases as l increases, in practice, for the purpose of saving computational load, (4.8) is approximated using the first two pairs of frequency sampling filters while neglecting the rest of the terms. That is

$$y(k) \approx T(e^{j\omega_d}) f^1(k) + T(e^{-j\omega_d}) f^{-1}(k) + T(e^{j3\omega_d}) f^3(k) + T(e^{-j3\omega_d}) f^{-3}(k) + v(k) \tag{4.9}$$

Define the complex parameter vector to be estimated as

$$\theta = [T(e^{j\omega_d})\ T(e^{-j\omega_d})\ T(e^{j3\omega_d})\ T(e^{-j3\omega_d})]^{T*}$$

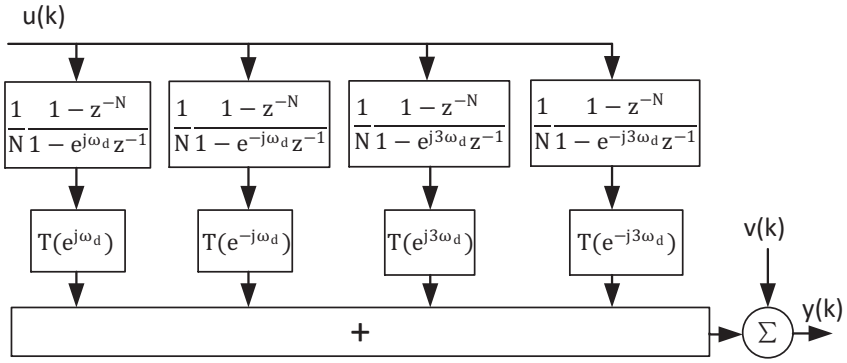


Figure 4.3: Block diagram of frequency sampling filter model using relay control.

and its corresponding regressor vector as

$$\phi(k) = [f^1(k)\ f^{-1}(k)\ f^3(k)\ f^{-3}(k)]^{T*}$$

where A^{T*} denotes the complex conjugate transpose of A.

The recursive least squares estimation algorithm is used to estimate the frequency response parameters, $T(e^{j\omega_d},\ T(e^{-j\omega_d}),\ T(e^{j3\omega_d}),\ T(e^{-j3\omega_d})$, under relay feedback control. Here, a standard recursive least squares algorithm is written as

$$P(k-1) = P(k-2) - \frac{P(k-2)^T\phi(k)\phi(k)^T P(k-2)}{1+\phi(k)^T P(k-2)\phi(k)} \tag{4.10}$$

$$\hat{\theta}(k) = \hat{\theta}(k-1) + P(k-1)\phi(k)(y(k) - \phi(k)^T\hat{\theta}(k-1)) \tag{4.11}$$

where $P(-1)$ and $\hat{\theta}(0)$ are the initial conditions selected for the recursive least squares algorithm. $\hat{\theta}(k)$ contains the estimated frequency response parameters. It should be noted that only the data segment in the steady state of the relay feedback is used for performing model identification. The data in the transient stage are ignored.

4.2.3 Estimation of Integrating Plus Delay Model

With the knowledge of the proportional controller K_p, the frequency response of the plant $G(e^{j\omega_d})$ is calculated from the closed-loop frequency response relationship,

$$T(e^{j\omega_d}) = \frac{G(e^{j\omega_d})K_p}{1 + G(e^{j\omega_d})K_p}$$

leading to

$$G(e^{j\omega_d}) = \frac{1}{K_p}\frac{T(e^{j\omega_d})}{1 - T(e^{j\omega_d})} \tag{4.12}$$

Note that the discrete-time frequency response $G(e^{j\omega_d})$ is a close approximation to its continuous-time counterpart under the assumption that the system operates in a fast sampling environment, where the equivalent continuous-time frequency is $\omega_1 = \frac{\omega_d}{\Delta t}$. Letting $G_p(j\omega_1)$ denote the plant continuous-time frequency response at the fundamental frequency ω_1, it follows that

$$G_p(j\omega_1) \approx G(e^{j\omega_d})$$

For an integrating plus time delay system, a single frequency is sufficient to determine its DC gain K and time delay d. The approximate model of an integrating system is assumed to be of the following form:

$$G_p(s) = \frac{Ke^{-ds}}{s} \tag{4.13}$$

For most physical systems, there are more or less approximations involved in obtaining the integrator plus time delay model.

Now, letting the frequency response of the integrator plus delay model (4.13) be equal to the estimated $G_p(j\omega_1)$ leads to

$$\frac{Ke^{-jd\omega_1}}{j\omega_1} = G_p(j\omega_1) \tag{4.14}$$

Equating the magnitudes on both side of (4.14) gives

$$K = \omega_1 |G_p(j\omega_1)| \tag{4.15}$$

where $|e^{-jd\omega_1}| = 1$. Additionally, from (4.14), the following relationship holds:

$$e^{-jd\omega_1} = \frac{j\omega_1 G_p(j\omega_1)}{K}$$

This gives the estimate of time delay as

$$d = -\frac{1}{\omega_1} tan^{-1} \frac{Imag(jG_p(j\omega_1))}{Real(jG_p(j\omega_1))} \tag{4.16}$$

It is seen here that if the system is approximated by an integrator with time delay, the plant information at a single frequency is sufficient to determine the plant gain and time delay.

4.2.4 PID controller design

Once the integrator plus time delay model is obtained, the PID controller parameters can be calculated using the tuning rules [Wang and Cluett, 1997b, 2000], which is introduced as follows.

Although there are many design methods available for integrator plus time delay model for instance, [Tyreus and Luyben, 1992; Visioli and Zhong, 2011], the set of tuning rules adopted here has the characteristics of being simple and robust, which was developed using frequency response analysis without approximation of time delay. With the tuning rules, the PID controller normalized parameters are calculated using the following equations:

$$\hat{K}_c = \frac{1}{0.5080\beta + 0.6208} \tag{4.17}$$

$$\hat{\tau}_I = 1.9885\beta + 1.2235 \tag{4.18}$$

$$\hat{\tau}_D = \frac{1}{1.0043\beta + 1.8194}. \tag{4.19}$$

The actual controller parameters are obtained by the following equations:

$$K_c = \frac{\hat{K}_c}{dK} \tag{4.20}$$

$$\tau_I = d\hat{\tau}_I \tag{4.21}$$

$$\tau_D = d\hat{\tau}_D. \tag{4.22}$$

The proportional, integral and derivative gains are calculated as:

$$K_p = K_c \tag{4.23}$$

$$K_i = \frac{K_c}{\tau_I} \tag{4.24}$$

$$K_d = K_c * \tau_D; \tag{4.25}$$

It should be noted that K_p calculated here is not the same as the proportional gain used in the relay experiments. For a larger β, a larger desired closed-loop time constant is selected, implying a slower closed-loop response speed for disturbance rejection and reference following. An advantage of the tuning rules is that the selection of β not only gives the desired closed-loop time constant, but also the gain margin and the phase margin for the closed-loop system [Wang and Cluett, 2000]. For instance, when $\beta = 2$, the gain margin and phase margin for a PID control system are approximately 3 and 45 degree, and a PI control system, 2 and 40 degree, respectively.

It should be emphasized that here, even though the time delay d and DC gain K are calculated by assuming a integrator plus delay model, they can approximately represent the dynamics of other high order integrating system. For example, if the actual plant model is a second order integrating system with time delay, the estimated time delay d is approximately the sum of the time delay in the actual plant plus the time constant of the first order dynamics in the plant. We will discuss this problem further in the following sections.

4.3 Numerical Simulation Validation of the Auto-tuning Process

In this section, two groups of simulation are conducted to validate the model identification and controller design stages in the auto-tuning process.

4.3.1 Validation of the integrator plus delay model

The first group of numerical simulations are conducted on three dynamics systems to validate the model identification stage of the auto-tuner.

The first plant is an integrator plus delay system:

$$G(s) = \frac{5e^{-0.032s}}{s} \tag{4.26}$$

The closed-loop relay experiment diagram is shown in Figure 4.4. The input and output

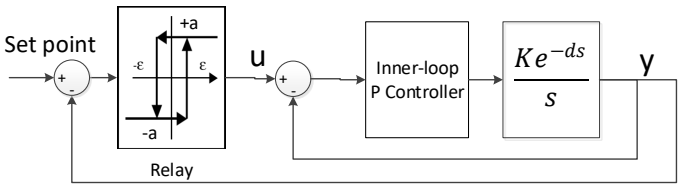


Figure 4.4: Relay feedback simulation 1 block diagram.

signals of the relay experiment are shown in Figure 4.5. The identified time delay is $0.0368s$ and the identified DC gain is 5.0023, both of which are very close to the actual parameters.

Practically speaking, the dynamic model can be more complicated than an integrator plus delay model. So the second simulation is carried out on a system with extra first order dynamics. The system's transfer function is given as

$$G(s) = \frac{109e^{-0.032s}}{s(0.072s + 1)} \tag{4.27}$$

The closed-loop relay experiment is shown in Figure 4.6. The input and output signals of the relay experiment are shown in Figure 4.7.

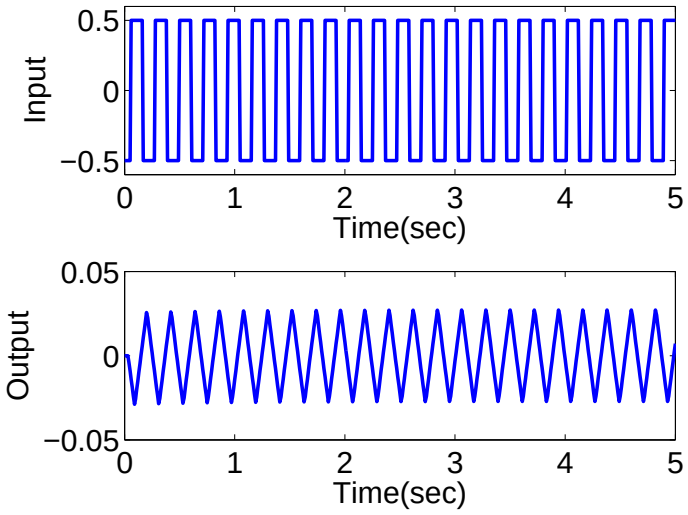


Figure 4.5: Relay feedback simulation 1: top figure input signal; bottom figure output signal.

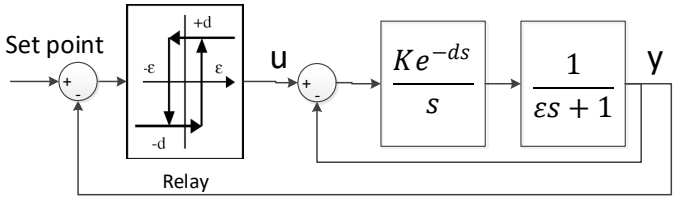


Figure 4.6: Relay feedback simulation 2 block diagram.

The identified time delay is $0.0917s$, and the identified DC gain is 75.1654. As the open-loop system in the second example has an extra first order dynamics, the identified time delay should be close to the transport delay plus the first order dynamics's time constant. The transport delay is 0.032 and the time constant is 0.072, so the sum of these two factors are close to what has been identified in the relay test.

The identified DC gain is smaller than the actual DC gain of the open-loop system. However, as the auto-tuning rule normally factors in a large gain margin, the difference between the actual and identified DC gains should not lead to serious performance degradation.

For many electromechanical systems, the cascade control structure is suitable. The common practice for such a cascade control structure is to tune the inner-loop sys-

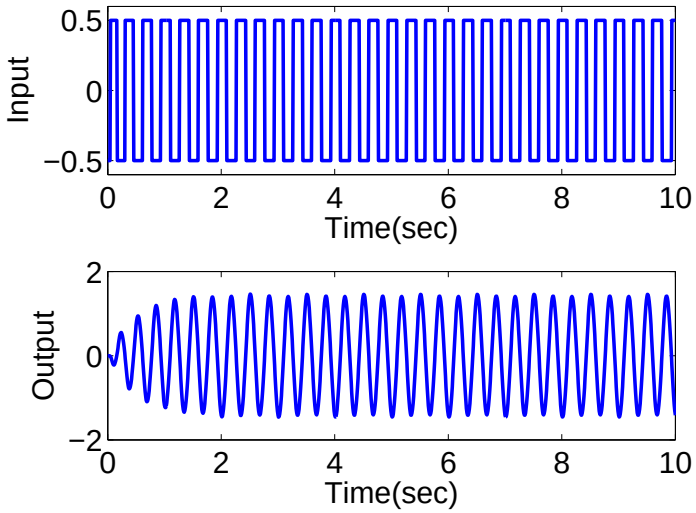


Figure 4.7: Relay feedback simulation 2: top figure input signal; bottom figure output signal.

tem to be much faster than the outer-loop system so that the inner-loop dynamic can be ignored. However, in some circumstances, the inner-loop system cannot be tuned fast enough and the outer-loop system is affected by the inner-loop dynamics. In this circumstance, the inner-loop dynamics needs to be taken into consideration when designing the outer-loop controller. In this simulation, identification is performed on a cascade control system to check if the identified model can represent both the inner-loop and the outer-loop dynamics.

The closed-loop relay experiment diagram is shown in Figure 4.8. The inner-loop

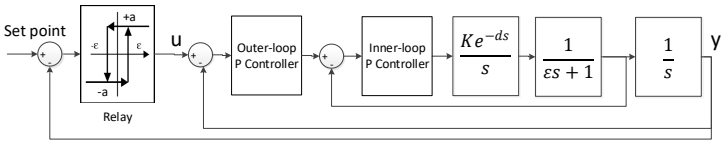


Figure 4.8: Relay feedback simulation 3 block diagram.

plant is a second-order integrating system with time delay and its transfer function is given as:

$$G_{in}(s) = \frac{109e^{-0.032s}}{s(0.072s+1)}. \tag{4.28}$$

. The inner-loop proportional gain is 0.4, so inner closed-loop transfer function is:

$$T_{in}(s) = \frac{43.6e^{-0.032s}}{0.072s^2 + s + 43.6}. \tag{4.29}$$

The outer-loop plant model is:

$$G_{out}(s) = G_{in}(s)\frac{1}{s} = \frac{43.6e^{-0.032s}}{s(0.072s^2 + s + 43.6)}. \tag{4.30}$$

Before the system identification result is given, the inner-loop step response is examined shown in Figure 4.9. The time constant of the inner closed-loop system is $0.253s$. The transport delay plus inner closed-loop time constant is $0.285s$. The inner-loop DC gain is 1. Then we compare these actual values with the identified results from the relay test.

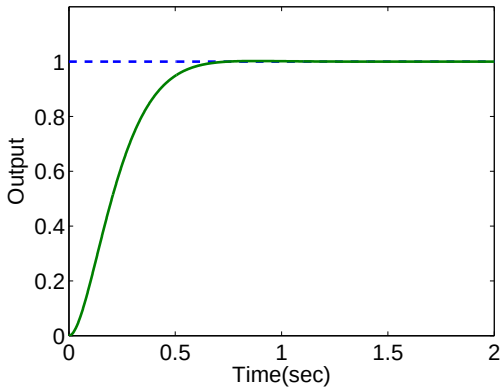


Figure 4.9: Inner closed-loop step response: dotted line set point;solid line output.

The input and output signal of the relay test is shown in Figure 4.10. The identified delay from the relay test is $0.24s$, which is slightly smaller than the actual value of time constant plus delay. The estimated DC gain is 0.8491, which is also smaller than the actual value of the DC gain. However, by having sufficient gain margin and phase margin in the outer-loop controller design, the outer-loop system's performance can still meet the design requirements.

From the above three simulation results, we can make the following conclusions. Integral systems with different levels of complexity can generally be fitted into the integrator plus delay model. The DC gain of the identified system is close to the actual

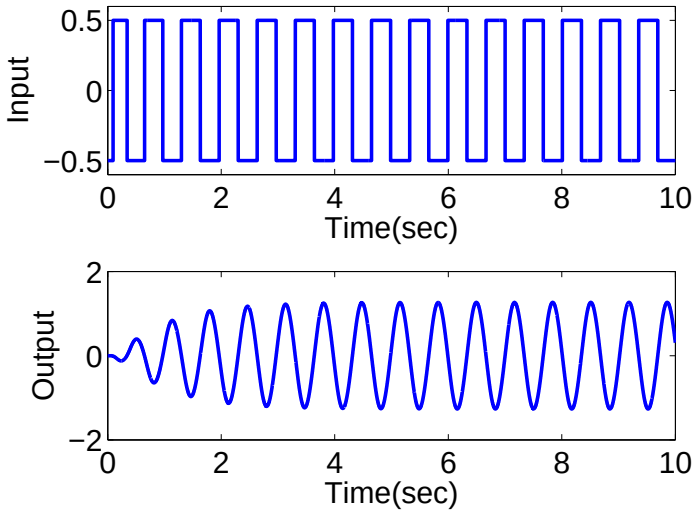


Figure 4.10: Relay feedback simulation 3: top figure input signal; bottom figure output signal.

system. The identified time delay is close to the actual system's pure time delay plus the time constant of other dynamics.

4.3.2 Auto-tuning application on two physical plants

In this section, we apply the auto-tuner on two physical plants. A question naturally related to the tuning results is: since the PID parameters are designed based on the simplified integrator plus delay model, will the actual closed-loop system obtain similar desired control performance? To answer this question, closed-loop frequency response tests are conducted on the actual plants and the simplified models.

The transfer function of the first plant is given as:

$$G(s) = \frac{549.8}{(0.072s + 1)s} e^{-0.032s} \tag{4.31}$$

As it can be seen, this system is a second order plus delay system with an integrator. From the relay test and system identification, we get the simplified integrator plus delay model as:

$$G(s) = \frac{343.256}{s} e^{-0.08740s}. \tag{4.32}$$

The next step is applying the tuning rule based PID design method on the identified integrator plus delay model. When β is chosen as 5, the PID parameters are: $K_p = 0.0105$, $K_i = 0.0108$ and $K_d = 0.000104$. Then we apply the obtained PID parameters on the actual system's transfer function (4.31) and the integrator plus delay transfer function (4.32). After that we perform frequency responses tests on both closed-loop systems. The frequency response results are shown in Figure 4.11. The solid line represents the frequency response of the actual closed-loop system and the dotted line is the frequency response of the closed-loop system with the simplified integrator plus delay model. As it can be seen two frequency response curves are similar to each other, which indicates that the closed-loop system's dynamics of the simplified model is quite close to the actual one.

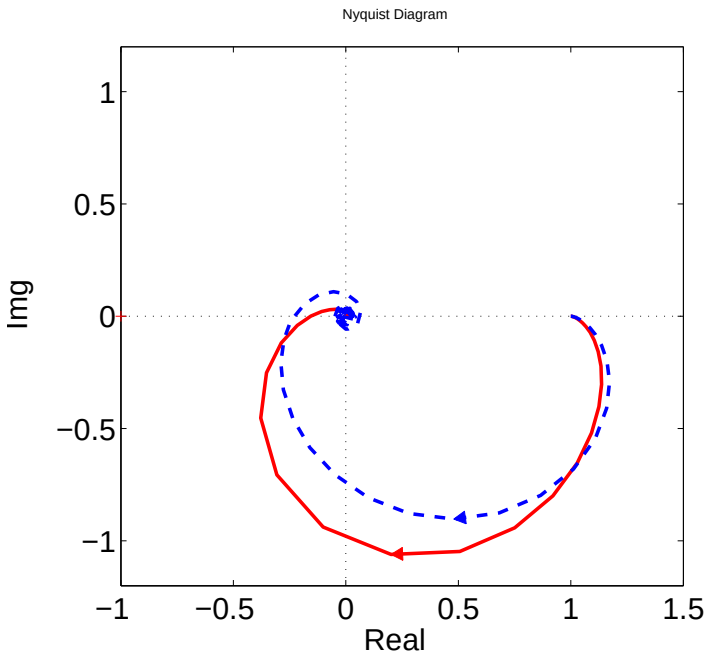


Figure 4.11: Closed-loop frequency response 1: dotted line frequency response of integrator plus delay system;solid line frequency response of the actual system.

Now we apply the auto-tuning method on another system:

$$G(s) = \frac{64}{s^2 + 16s + 64} \frac{1}{0.2s + 1} \frac{1}{s} e^{-0.032s}. \tag{4.33}$$

This is a fourth-order system with an integrator. From the relay test, we get the simplified integrator plus delay model as:

$$G(s) = \frac{0.7128}{s} e^{-0.4477}. \tag{4.34}$$

When β is chosen as 10, the PID parameters are: $K_p = 0.5496$, $K_i = 0.0852$ and $K_d = 0.0156$. Then we apply the obtained PID parameters on the actual system's transfer function (4.33) and the integrator plus delay transfer function (4.34), and then perform frequency responses tests on both closed-loop systems. The frequency response results are shown in Figure 4.12. The solid line represents the frequency response of the actual closed-loop system and the dotted line is the frequency response of the closed-loop system with the integrator plus delay model. As it can be seen two frequency response curves are close to each other, which indicates that the closed-loop system's dynamics of the identified model is quite close to the actual one. Furthermore, the closed-loop step responses are shown in Figure (4.12). As we can see, the actual closed-loop step response and the expected closed-loop step response are similar to each other.

From the above two simulations, it can be concluded that even though the auto-tuning process adopts the simplified integrator plus delay model to design the PID controller, the resultant PID parameters can actually be applied on more complex high order systems to give robust control performance. The reason is the open-loop dynamics is dominated by the integrator and the time delay.

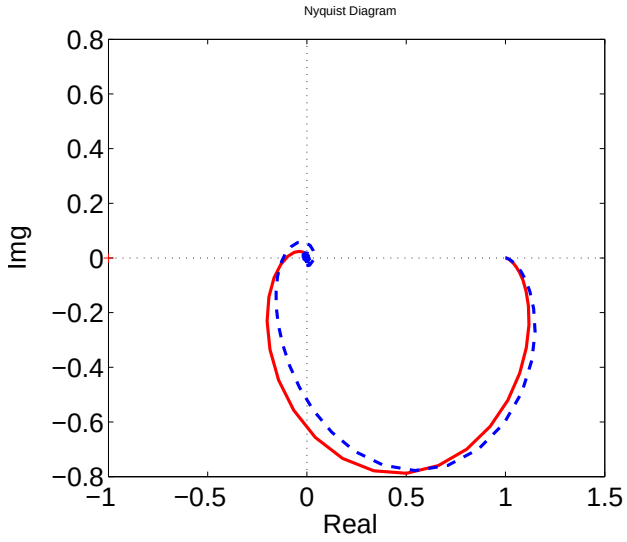


Figure 4.12: Outer closed-loop frequency response 2: dotted line frequency response of integrator plus delay system;solid line frequency response of the actual system.

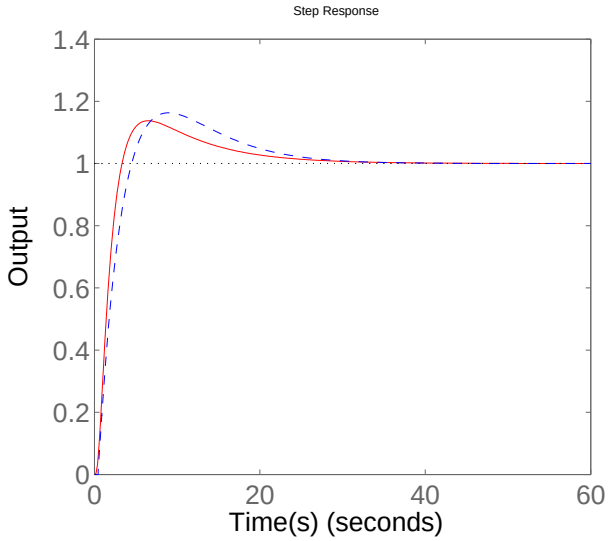


Figure 4.13: Outer closed-loop step response 2: dotted line step response of integrator plus delay system;solid line step response of the actual system.

4.4 Experimental Validation

Figure 4.14: Quadrotor TC3.0 test-bed

4.5 Auto-tuning of Quadrotor TC1.0

In order to conduct identification experiments in a controlled environment, the quadrotor is required to be fixed on a mechanical stand. The test-bed shown in Figure 4.15 is built to conduct the relay experiments for identification of the quadrotors' x and y axis dynamics. For example, when identifying the roll angle dynamics, quadrotor arms along the x axis are fixed on the stand so that the quadrotor can only rotate about the x axis. The test-bed is carefully adjusted to make rotating axis aligned with the quadrotor's body frame axis, so that the torque due to weight force is minimized. Besides, as the quadrotor platform is very light, and the two rotating pivots are very smooth, the friction is negligible. A custom control firmware is uploaded to the microprocessor to make the quadrotor rotate under relay control at a certain frequency. The sampling time for the inner-loop, outer-loop controllers and relay test are all set to be $0.01s$, which is the IMU sensor's maximum updating rate. In this section, we apply the auto-tuning process on TC1.0. In order to be consistent with the model-based controller design described in Chapter 3, PI controllers are used on both inner-loop and outer-loop. However, as it will be discussed in the next TC3.0 controller tuning section, other controller configurations could offer better results.

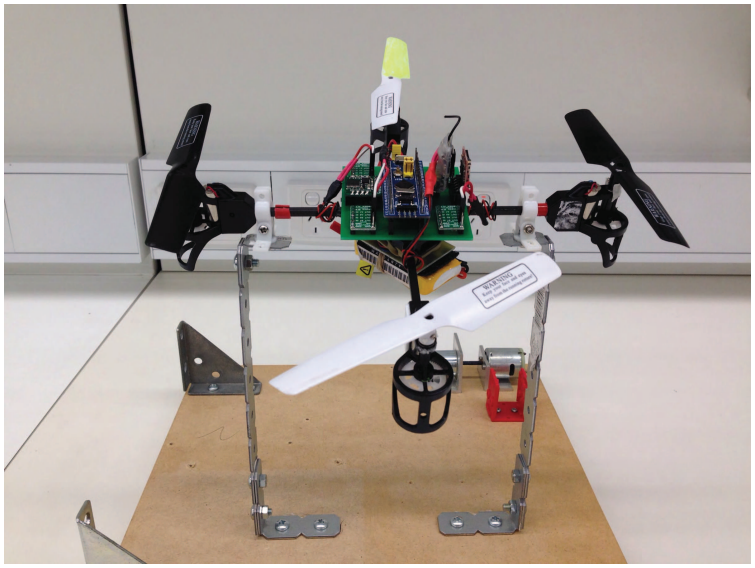


Figure 4.15: Quadrotor TC1.0 test-bed

4.5.1 Auto-tuning of TC1.0 Secondary PI Controller

Before the relay control experiment, a proportional controller $K_{T_1} = 0.06$ is selected to stabilize the secondary system with sampling interval $\Delta t = 0.01$ (sec). The sensor measurement of velocity, $\dot{\theta}(t)$, contains noise. Thus, a relay amplitude of 0.8 together with a hysteresis level of 0.1 is selected to reflect the measurement noise level. Figure 4.16 shows a segment of the relay feedback control data. From the input signal to the inner-loop closed-loop control system, the period of the sustained oscillations is identified as $N = 28$ samples, leading to the fundamental frequency in the frequency sampling filter as $\frac{2\pi}{N}$. The use of the frequency sampling filter based estimation algorithm gives the inner-loop closed-loop frequency response as

$$T(e^{j\frac{2\pi}{N}}) = -1.225 - j0.5137$$

Converting this discrete-time frequency to continuous time frequency, which is $\omega_1 = \frac{2\pi}{N\Delta t} = 22.44$, together with the knowledge of the proportional controller used in the relay experiment ($K_{T_1} = 0.06$), the continuous-time frequency response of the inner-

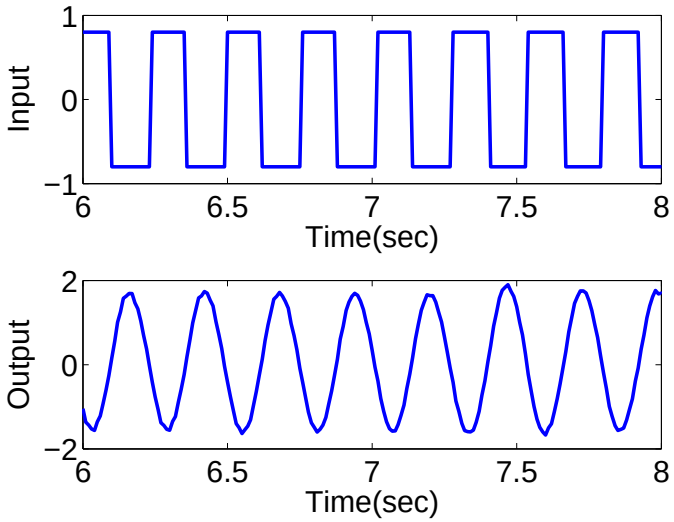


Figure 4.16: TC1.0 Relay feedback control signals from inner-loop system: top figure input signal; bottom figure output signal.

loop plant is calculated as

$$G_1(j\omega_1) = \frac{1}{K_{T_1}} \frac{T(e^{j\frac{2\pi}{N}})}{1 - T(e^{j\frac{2\pi}{N}})} = -9.551 - j1.6421$$

From this frequency information, an integrating plus delay model is identified as

$$G_1(s) = \frac{217.5589e^{-0.0624s}}{s}$$

By choosing the desired closed-loop time constant as $\tau_{cl} = \beta d = 3d = 0.1872$ (sec) ($\beta = 3$) and damping coefficient of 1, the PI controller parameters are found for the inner-loop control system as $K_c = 0.0343$ and $\tau_I = 0.4487$. This set of PI controller parameters approximately gives a gain margin of 3 and phase margin of 48 degree for the closed-loop system with the integrator plus delay model. Figure 4.17 shows the closed-loop step response of $\dot{\theta}(t)$, where the reference signal has a magnitude of 0.5 (rad/sec). It is seen from this figure that the closed-loop velocity response follows the reference signal without any steady-state error, and there is a large overshoot and a slight oscillation. Additionally, there are disturbances and measurement noise in the inner-loop system. For a cascade control system, the inner-loop control system is

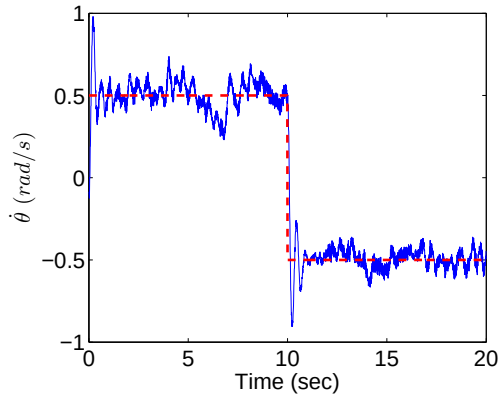


Figure 4.17: Inner-loop step response in closed-loop control. Dashed line: reference signal; solid line: output.

required to have a fast response speed, which is achieved in the design here by choosing a smaller β value.

4.5.2 Auto-tuning of TC1.0 Primary PI Controller

The proportional controller used for the relay control experiment is selected as $K_{T_2} = 2$. The amplitude of the relay is 0.3 and the hysteresis level ϵ is 0.05 to prevent the relay from random switching. Figure 4.18 shows a segment of the input and output data generated from this relay feedback control. The averaged period of the sustained oscillation is $N = 59$ in number of samples, which gives the fundamental frequency in discrete-time as $\frac{2\pi}{59}$. Frequency sampling filters model is used to estimate the closed-loop frequency response based on the set of input and output data shown in Figure 4.18, yielding to

$$T(e^{j\frac{2\pi}{N}}) = -0.1317 - j0.3130$$

With the proportional controller $K_{T_2} = 2$, the frequency response of the outer-loop system is found at $\omega_1 = \frac{2\pi}{N\Delta t} = 14.2$ as

$$G_2(j\omega_1) = \frac{1}{K_{T_2}} \frac{T(e^{j\frac{2\pi}{N}})}{1 - T(e^{j\frac{2\pi}{N}})} = -0.0896 - j0.1135$$

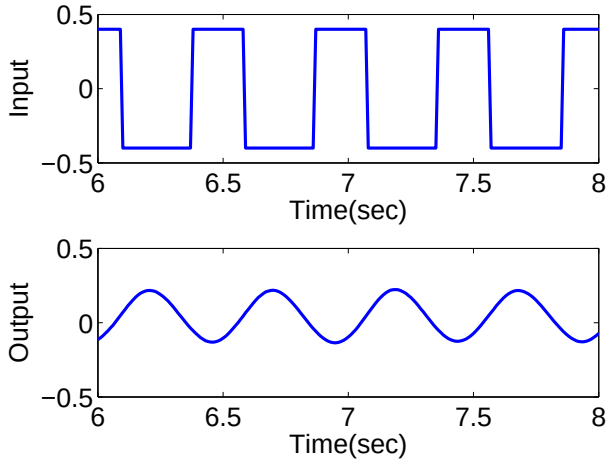


Figure 4.18: TC1.0 relay feedback control signals from outer-loop system: top figure input signal; bottom figure output signal.

From this frequency information, the integrator plus delay model for the primary system is calculated as

$$G_2(s) = \frac{1.54e^{-0.0627s}}{s}$$

For comparison purpose, when a faster desired closed-loop time constant $\tau_{cl} = 5d$ ($\beta = 5$, gain margin ≈ 5, and phase margin ≈ 55 degree) and a slower desired closed-loop time constant $\tau_{cl} = 10d$ ($\beta = 10$, gain margin ≈ 9, and phase margin ≈ 65 degree) are used in the computation, the PI controller parameters become $K_c = 3.27$, $\tau_I = 0.7006$, and $K_c = 1.8155$, $\tau_I = 1.3244$. Figure 4.19 shows the comparative closed-loop responses for the three cases experimentally. It is seen from the comparative results that all three PI controllers lead to stable closed-loop systems. Clearly when $\beta = 5$, the fastest outer-loop response is obtained.

4.5.3 TC1.0 Flight Testing Results

To evaluate the control performance in the actual flying conditions, experiments are conducted. The PI controller parameters achieved from the auto-tuner are tested on TC1.0. Before the flight test, the controller parameters calculated by the auto-tuner are uploaded to the quadrotor's on-board microprocessor. During the flight test, the reference signals ϕ^*, θ^* and $\dot{\psi}^*$ are manually changed using a RC transmitter. These

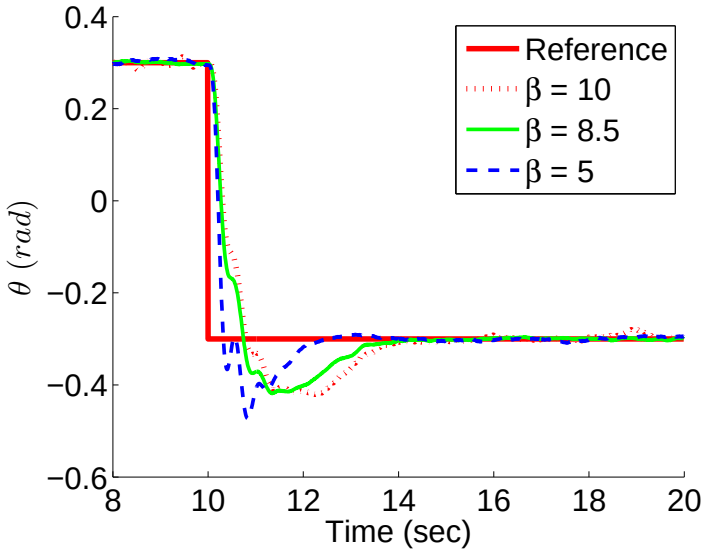
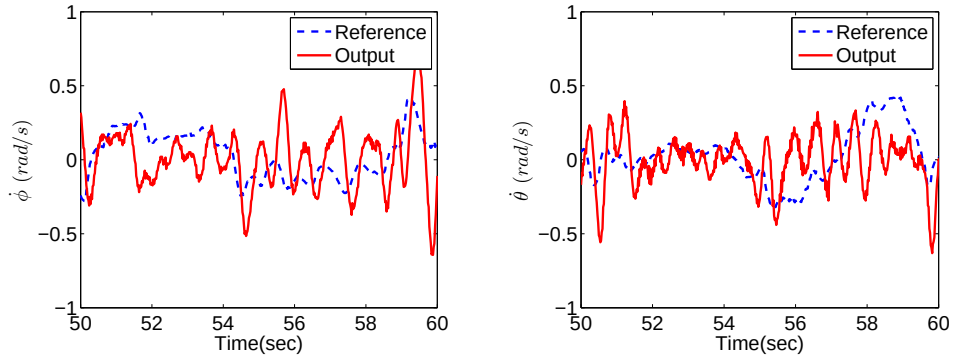


Figure 4.19: Comparative outer-loop step response in closed-loop control.

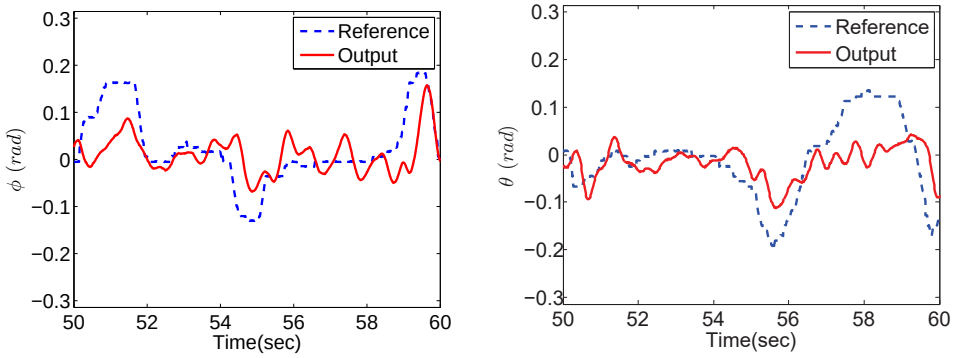
reference signals are then received by the RC receiver on the quadrotor and sent to the on-board microprocessor. The microprocessor compares the difference between the reference signals from the receiver and the actually feedback signals from the sensors and generate the manipulated variables to stabilize the quadrotor system. The quadrotor's data logger records the flight test data including reference signals, feedback signals, manipulated variables, sampling time and the controller parameters.

Figures 4.20(a)-(b) show the inner-loop control system responses to the references at the flight testing environment. Figures 4.21(a)-(b) show the outer-loop control system responses to reference changes. In Figure 4.21 we can see that the Euler roll and pitch angles of TC1.0 can follow the reference signals but errors exist. It should be noted the actual performance is not only determined by the controller tuning but also limited by the hardware. Compared with flying test results shown in chapter 3, it can be concluded that auto-tuner already offers the best control performance that the specific quadrotor hardware can offer.



(a) Flight roll rate (b) Flight pitch rate

Figure 4.20: TC1.0 Inner-loop control system responses to reference changes. Key: dotted line-reference; solid line- measured output



(a) Flight roll angle (b) Flight pitch angle

Figure 4.21: TC1.0 Outer-loop control system responses to reference changes. Key: dotted line-reference; solid line- measured output

4.5.4 Comparison with SIMC PI tuning rules

In this section, we compare the auto-tuner PI tuning rules with another method called SIMC PID tuning rules [Skogestad, 2003], which is also based on the simplified plant model. SIMC tuning rules are described as follows. For an integrating with delay system

$$g(s) = \frac{Ke^{-ds}}{s} \tag{4.35}$$

The proportional gain K_c and integral time constant τ_I are calculated as:

$$K_c = \frac{1}{K} * \frac{1}{\tau_c + d} \tag{4.36}$$

$$\tau_I = 4(\tau_c + d) \tag{4.37}$$

where τ_c is the desired closed-loop time constant and a tuning parameter for the closed-loop performance. For fast response and good robustness, τ_c should be equal to the time delay d.

Now we will firstly compare the TC1.0 inner-loop PID tuning results when using the two different tuning rules. The inner-loop plant's DC gain $K = 217.5589$ and time delay $d = 0.0624$. The PI controller parameters calculated by the two tuning rules are listed in Table 4.1. The left column lists the auto-tuner's PI parameters, when the tuning factor β varies from 1 to 5 and the closed-loop time constant tau_c varies from d to $5d$ accordingly. The right column lists the SIMC PI parameters, when the desired closed-loop time constant tau_c varies from d to $5d$. Apparently, when choosing the same closed-loop time constant, the SIMC tuning gives smaller K_c and larger tau_I, which lead to slower closed-loop responses. When tau_c in SIMC is chosen to be d and β in the auto-tuner is chosen to be $3d$, these two tuning rules generate close K_c and tau_I.

Table 4.1: Auto-tuner and SIMC PI tuning rules comparison on TC1.0 inner-loop

Auto-tuner	**SIMC**
$K_c = 0.0653\ \tau_I = 0.2004\ (\beta = 1, \tau_c = d)$	$K_c = 0.0368\ \tau_I = 0.4992\ (\tau_c = d)$
$K_c = 0.0450\ \tau_I = 0.3245\ (\beta = 2, \tau_c = 2d)$	$K_c = 0.0246\ \tau_I = 0.7488\ (\tau_c = 2d)$
$K_c = 0.0343\ \tau_I = 0.4486\ (\beta = 3, \tau_c = 3d)$	$K_c = 0.0184\ \tau_I = 0.9984\ (\tau_c = 3d)$
$K_c = 0.0233\ \tau_I = 0.6968\ (\beta = 5, \tau_c = 5d)$	$K_c = 0.0123\ \tau_I = 1.4976\ (\tau_c = 5d)$

The comparison between the two tuning rules are also conducted on the TC1.0 outer-loop. The outer-loop plant's DC gain $K = 1.54.$ and time delay $d = 0.0627$. The PI controller parameters calculated by the two tuning rules are listed in Table 4.2. The left column lists the auto-tuner's PI parameters, when the tuning factor β varies from 5 to 10 and the closed-loop time constant tau_c varies from $5d$ to $10d$ accordingly. The right column lists the SIMC PI parameters, when the desired closed-loop time constant tau_c varies from $5d$ to $10d$. Similar to the inner-loop, when choosing the same closed-loop time constant, the SIMC tuning gives smaller K_c and larger tau_I,

which lead to slower closed-loop responses. When tau_c in SIMC is chosen to be $5d$, and β in the auto-tuner is chosen to be $10d$, these two tuning rules generate close K_c and tau_I.

Table 4.2: TAB ch6 Auto-tuner and SIMC PI tuning rules comparison on TC1.0 outer-loop

Auto-tuner	**SIMC**
$K_c = 3.2765\ \tau_I = 0.7001\ (\beta = 5, \tau_c = 5d)$	$K_c = 1.7261\ \tau_I = 1.5048\ (\tau_c = 5d)$
$K_c = 2.4795\ \tau_I = 0.9495\ (\beta = 7, \tau_c = 7d)$	$K_c = 1.2946\ \tau_I = 2.0064\ (\tau_c = 7d)$
$K_c = 2.2107\ \tau_I = 1.0741\ (\beta = 8, \tau_c = 8d)$	$K_c = 1.1507\ \tau_I = 2.2572\ (\tau_c = 8d)$
$K_c = 1.8167\ \tau_I = 1.3235\ (\beta = 10, \tau_c = 10d)$	$K_c = 0.9415\ \tau_I = 2.7588\ (\tau_c = 10d)$

From the above comparisons, it can be concluded the auto-tuner PI tuning rules generally offer faster closed-loop response than SIMC tuning rules. However, by choosing different closed-loop time constants, both tuning rules can offer similar closed-loop control performance.

4.6 Auto-tuning of Quadrotor TC3.0

The same auto-tuning process is carried out on TC3.0. Different PID controller configurations are compared. Figure 4.14 is TC3.0's relay test-bed, which share the similar design principle as TC1.0 with some improvement. The major improvement it the quadrotor body is higher from the ground, which minimizes the ground effect (the deflected air disturbance from the ground).

4.6.1 Auto-tuning of TC3.0 Secondary PID Controller

Before the relay control experiment, a proportional controller $K_{T_1} = 0.03$ is selected to stabilize the secondary system with sampling interval $\Delta t = 0.01s$. The sensor measurement of velocity, $\dot{\phi}(t)$, contains noise. Thus, a relay amplitude of $1rad/s$ together with a hysteresis level of $0.1rad$ is selected to reflect the measurement noise level. Figure 4.22 shows a segment of the relay feedback control data. From the input signal to the inner-loop closed-loop control system, the period of the sustained oscillations is identified as $N = 28$ samples, leading to the fundamental frequency in

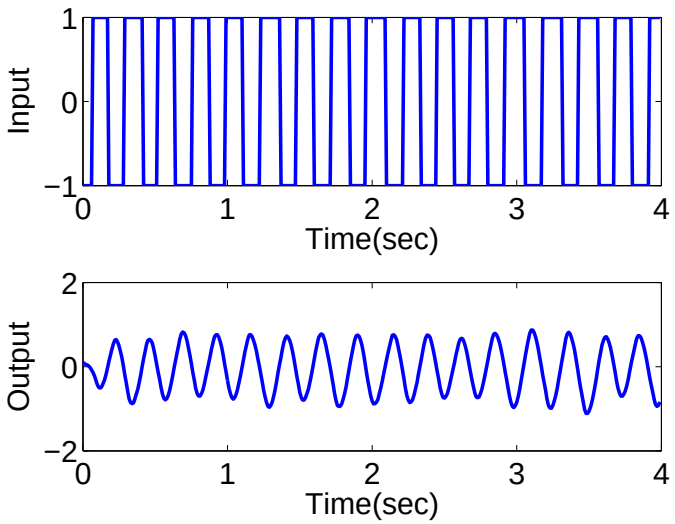


Figure 4.22: TC3.0 Relay feedback control signals from inner-loop system: top figure input signal; bottom figure output signal.

.

the frequency sampling filter as $\frac{2\pi}{N}$. The use of the frequency sampling filter based estimation algorithm gives the inner-loop closed-loop frequency response as

$$T(e^{j\frac{2\pi}{N}}) = -0.4947 - j0.3794$$

Converting this discrete-time frequency to continuous time frequency, which is $\omega_1 = \frac{2\pi}{N\Delta t} = 22.44$, together with the knowledge of the proportional controller used in the relay experiment ($K_{T_1} = 0.03$), the continuous-time frequency response of the inner-loop plant is calculated as

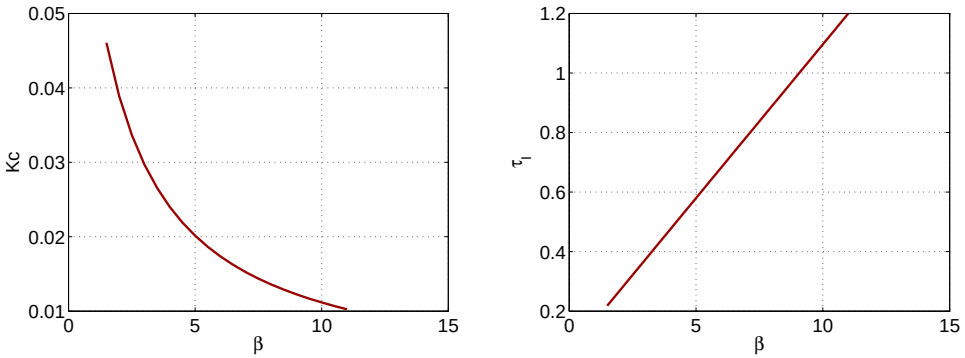
$$G_1(j\omega_1) = \frac{1}{K_{T_1}} \frac{T(e^{j\frac{2\pi}{N}})}{1 - T(e^{j\frac{2\pi}{N}})} = -12.3824 - j5.3181$$

From this frequency information, an integrating plus delay model is identified as

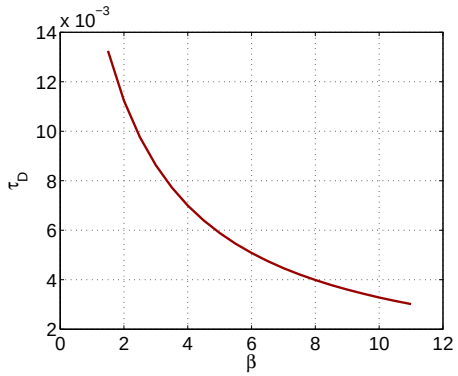
$$G_1(s) = \frac{302.4e^{-0.0519s}}{s}$$

The PID parameters with β changing from 1 to 11 are calculated and plotted in

Figure 4.23. In order to compare the stability margins of PID, PI and PD controllers with β changing from 1 to 11, Nyquist loci together with gain margins and phase margins for different PID controller structures are calculated.



(a) K_c (b) τ_I



(c) τ_D

Figure 4.23: TC3.0 Inner-loop PID autotuning parameters

Inner-loop PID: The Nyquist diagrams of TC3.0 inner-loop with PID controller are plotted in Figure 4.24. The gain margins and phase margins for the same system are plotted in Figure 4.25. When β varies between 1.5 and 11, the gain margin varies between 2.49 to 10.16 and the phase margin varies between 40.23 degree and 66.39 degrees.

Inner-loop PD: The Nyquist diagrams of TC3.0 inner-loop with PD controller are plotted in Figure 4.26. The gain margins and phase margins for the same system are plotted in Figure 4.27. When β varies between 1.5 and 11, the gain margin varies between 2.51 to 10.3 and the phase margin varies between 58.47 degree and 81.31

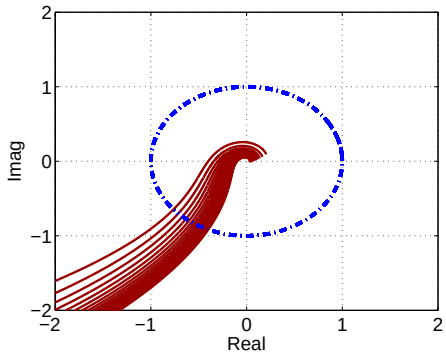
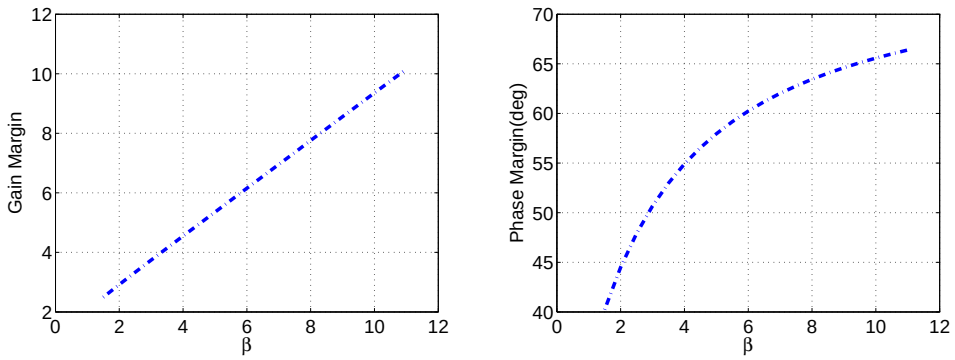


Figure 4.24: TC3.0 inner-loop with PID controller Nyquist plot



(a) gain margin (b) phase margin

Figure 4.25: TC3.0 inner-loop with PID controller stability margins

degree.

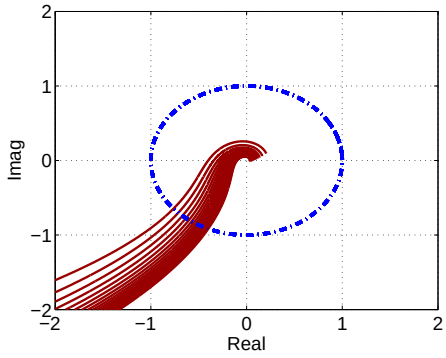
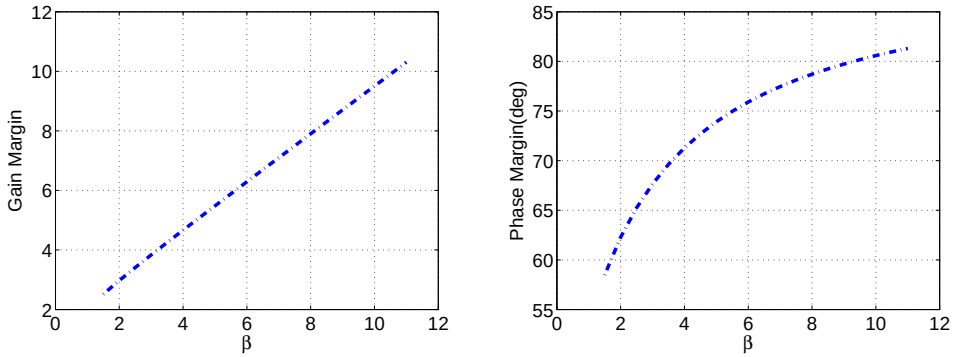


Figure 4.26: TC3.0 inner-loop with PD controller Nyquist plot



(a) gain margin (b) phase margin

Figure 4.27: TC3.0 inner-loop with PD controller stability margins

Inner-loop PI: The Nyquist diagrams of TC3.0 inner-loop with PI controller are plotted in Figure 4.28. The gain margins and phase margins for the same system are plotted in Figure 4.29. When β varies between 1.5 and 11, the gain margin varies between 1.91 to 9.57 and the phase margin varies between 29.16 degree and 65.88 degree.

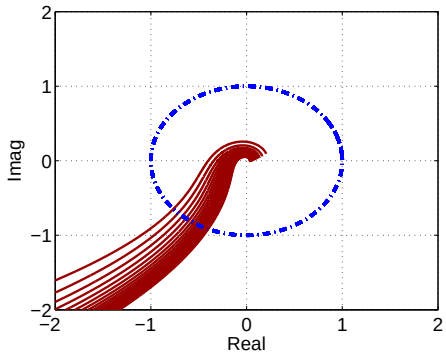
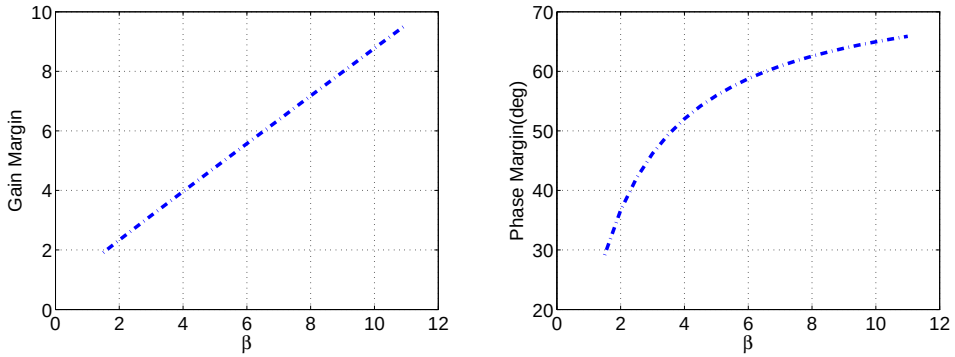


Figure 4.28: TC3.0 inner-loop with PI controller Nyquist plot

The gain margins and phase margins of the inner-loop system with different types of controllers are summarized in Table 4.3. Apparently, the inner-loop system with PD controller has the largest gain margin and phase margin. Furthermore, it is not necessary to use integral controller to eliminate steady-state errors. For these reasons, PD controller should offer the best control performance for the quadrotor inner-loop system.



(a) gain margin (b) phase margin

Figure 4.29: TC3.0 inner-loop with PI controller stability margins

Table 4.3: Quadrotor TC3.0 inner-loop stablity margins

	gain margin	**phase margin(deg)**
PID	$2.19 - 10.16$	$40.23 - 66.39$
PI	$1.91 - 9.57$	$29.16 - 65.88$
PD	$2.51 - 10.3$	$51.47 - 81.31$

4.6.2 Auto-tuning of TC3.0 Primary PID Controller

The proportional controller used for the relay control experiment is selected as $K_{T_2} = 4$. The amplitude of the relay is 0.3 and the hysteresis level ϵ is 0.1 to prevent the relay from random switching. Figure 4.30 shows a segment of the input and output data generated from this relay feedback control. The averaged period of the sustained oscillation is $N = 44$ in number of samples, which gives the fundamental frequency in discrete-time as $\frac{2\pi}{44}$. Frequency sampling filters model is used to estimate the closed-loop frequency response based on the set of input and output data shown in Figure 4.30, yielding to

$$T(e^{j\frac{2\pi}{N}}) = -0.3536 - j0.3487$$

With the proportional controller $K_{T_2} = 4$, the frequency response of the outer-loop system is found at $\omega_1 = \frac{2\pi}{N\Delta t} = 14.28$ as

$$G_2(j\omega_1) = \frac{1}{K_{T_2}} \frac{T(e^{j\frac{2\pi}{N}})}{1 - T(e^{j\frac{2\pi}{N}})} = -0.0768 - j0.0446$$

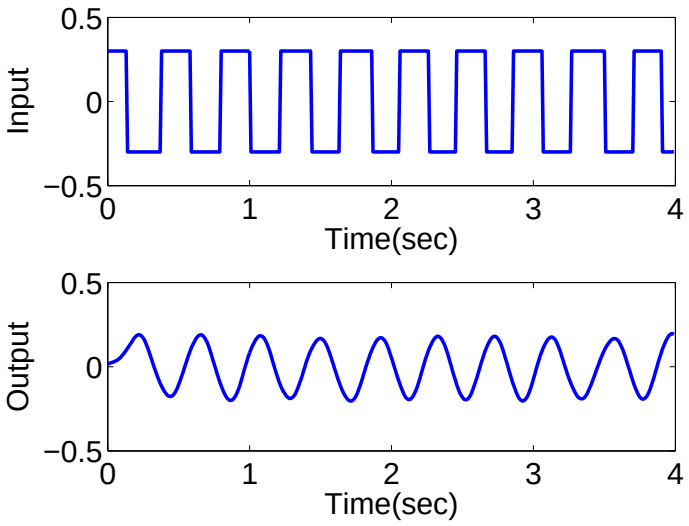


Figure 4.30: TC3.0 Relay feedback control signals from outer-loop system: top figure input signal; bottom figure output signal.

From this frequency information, the integrator plus delay model for the primary system is calculated as

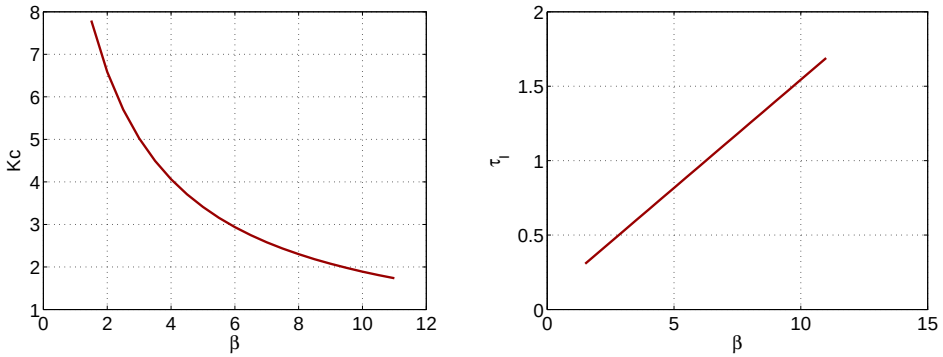
$$G_2(s) = \frac{1.26e^{-0.0731s}}{s}$$

The PID parameters with β changing from 1 to 11 are calculated and plotted in Figure 4.31. In order to compare the stability margins of PID, PI and PD controllers with β changing from 1 to 11, the Nyquist diagrams, gain margins and phase margins for different PID controller structures are given.

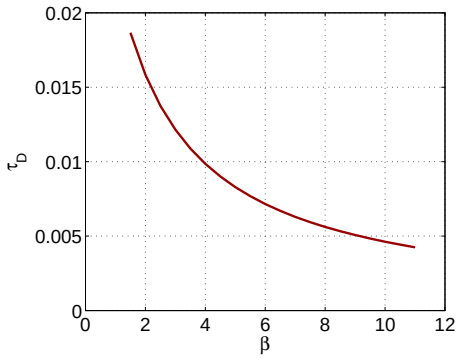
Outer-loop PID: The Nyquist diagrams of TC3.0 Outer-loop with PID controller are plotted in Figure 4.32. The gain margins and phase margins for the same system are plotted in Figure 4.33. When β varies between 1.5 and 11, the gain margin varies between 2.49 to 10.16 and the phase margin varies between 40.23 degree and 66.39 degree.

Outer-loop PD: The Nyquist loci of TC3.0 Outer-loop with PD controller are plotted in Figure 4.34. The gain margins and phase margins for the same system are plotted in Figure 4.35. When β varies between 1.5 to 11, the gain margin varies between 2.51 to 10.3 and the phase margin varies between 58.47 degree to 81.31 degree.

Outer-loop PI: The Nyquist diagrams of TC3.0 Outer-loop with PI controller are



(a) K_c (b) τ_I



(c) τ_D

Figure 4.31: TC3.0 Outer-loop PID autotuning parameters

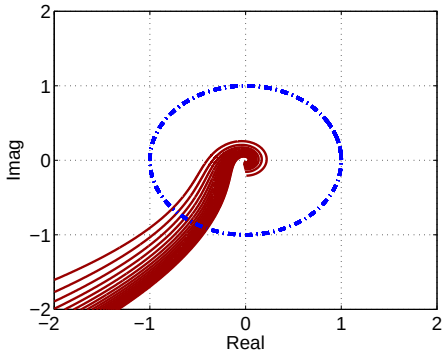
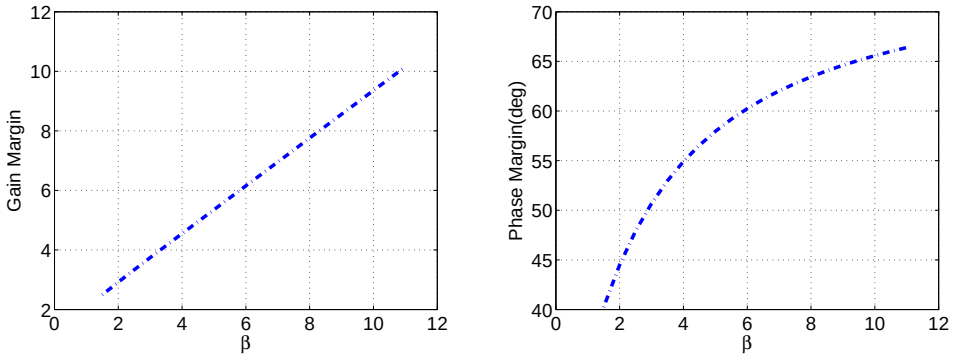


Figure 4.32: TC3.0 Outer-loop with PID controller Nyquist plot

plotted in Figure 4.36. The gain margins and phase margins for the same system are plotted in Figure 4.37. When β varies between 1.5 and 11, the gain margin varies



(a) gain margin (b) phase margin

Figure 4.33: TC3.0 Outer-loop with PID controller stability margins

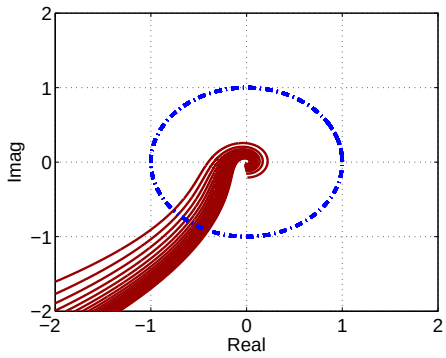
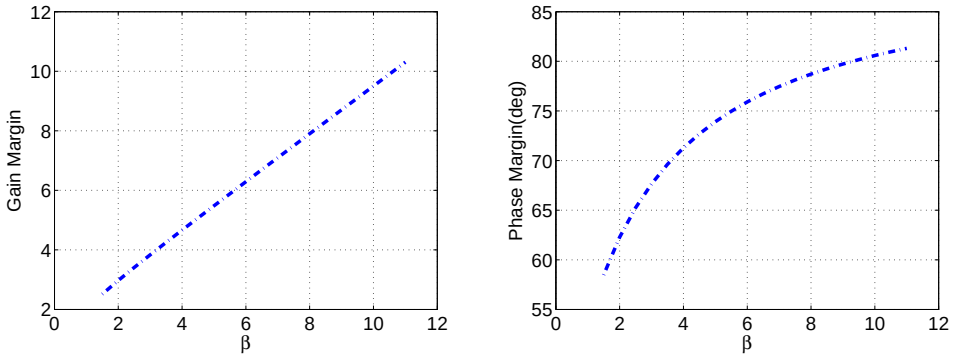


Figure 4.34: TC3.0 Outer-loop with PD controller Nyquist plot



(a) gain margin (b) phase margin

Figure 4.35: TC3.0 Outer-loop with PD controller stability margins

between 1.91 to 9.57 and the phase margin varies between 29.16 degree and 65.88 degree.

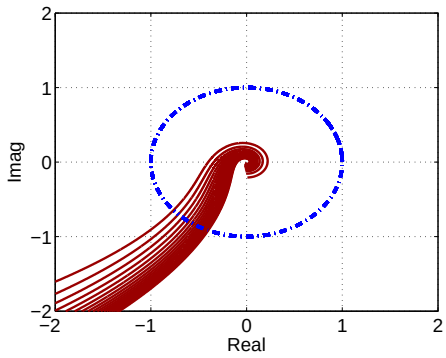
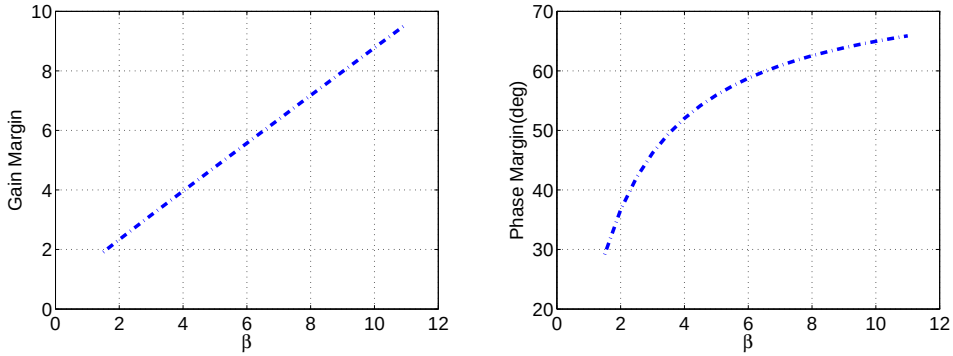


Figure 4.36: TC3.0 Outer-loop with PI controller Nyquist plot



(a) gain margin (b) phase margin

Figure 4.37: TC3.0 Outer-loop with PI controller stability margins

The gain margins and phase margins of the outer-loop system with different types of controllers are summarized in Table 4.4. In order to eliminate steady-state errors in the outer-loop system, integral controllers must be used. Furthermore, PID controller offer larger gain margin and phase margin than PI controller, so PID controller is chosen as the outer-loop controller.

TC3.0's step responses are given in Figure 4.38. The inner-loop controllers used in these tests are PD controllers. For the outer-loop, PID, PD and PI controllers are used in three step response tests. As it can be seen, the systems with PID and PI

Table 4.4: Quadrotor TC3.0 outer-loop stability margins

	gain margin	phase margin(deg)
PID	$2.19 - 10.16$	$40.23 - 66.39$
PI	$1.91 - 9.57$	$29.16 - 65.88$
PD	$2.51 - 10.3$	$51.47 - 81.31$

controllers have very similar performance with no steady-state error. The system with PD controller shows steady-state error but it has the smallest overshot among three systems.

To further evaluate the performance of difference outer-loop controllers, mean squared errors (MSE) of the three step response tests are calculated using the following equation:

$$MSE = \frac{\sum_{n=1}^{M}(r(n) - y(n))^2}{M}, \tag{4.38}$$

where M is the number of samples, r is the setpoint and y is the output. Among the three controllers, PID controlled system gives the lowest MSE (0.0038) and the MSE of PI and PD controlled systems are 0.0040 and 0.0052 respectively.

Table 4.5: TC3.0 inner-loop and outer-loop controller parameters

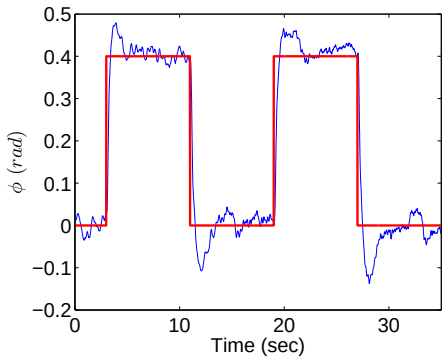
	K_p	K_i	K_d
Inner-loop	0.042	0	0.00045
Outer-loop	3.22	2.87	0.023

4.6.3 TC3.0 Flight Testing Results

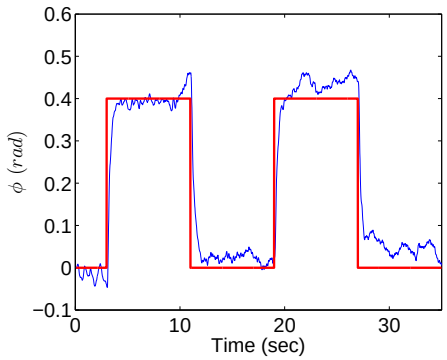
Flight tests are also conducted on quadrotor TC3.0. The controller parameters achieved from the auto-tuner are given in Table 4.5.

Figures 4.39(a)-(c) show the inner-loop control system responses to the references at the flight testing environment. Figures 4.40(a)-(b) show the outer-loop control system responses to reference changes.

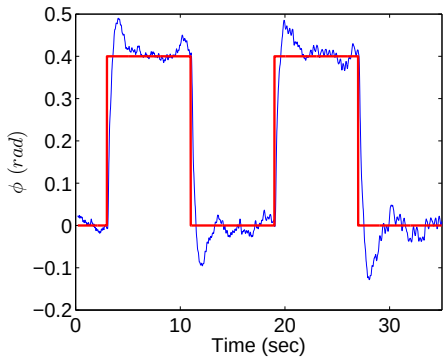
Compared with the angle tracking performance of TC1.0, Figure 4.40 shows TC3.0 has a superior closed-loop control performance. This is because TC3.0 is equipped with more powerful actuators and the overall design is improved. For both TC1.0 and TC3.0, the auto-tuner helps them to reach the specific hardware's best control performance.



(a) PI

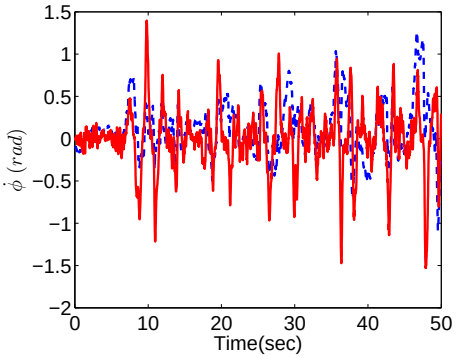


(b) PD

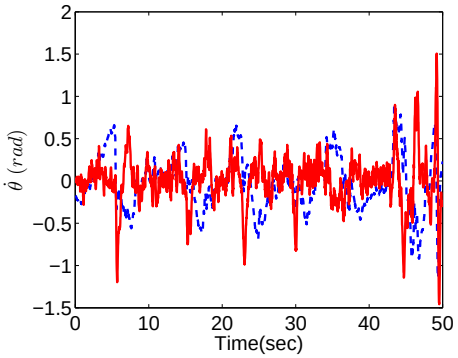


(c) PID

Figure 4.38: TC3.0 Outer-loop step responses. Key: dotted line-reference; solid line-measured output

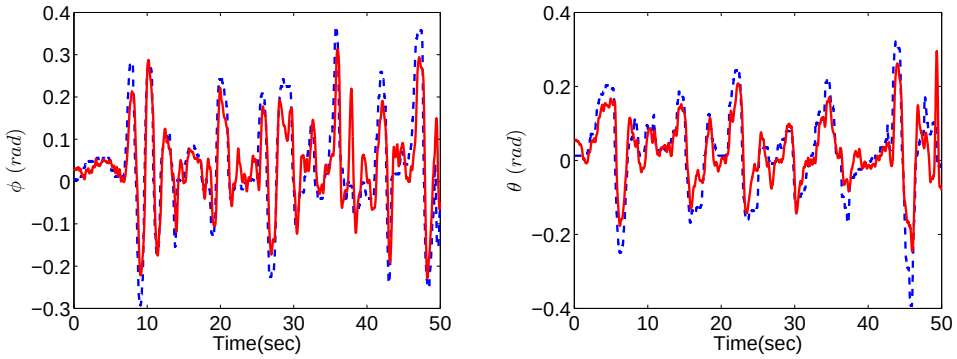


(a) Flight roll rate



(b) Flight pitch rate

Figure 4.39: TC3.0 inner-loop control system responses to reference changes. Key: dotted line-reference; solid line- measured output



(a) Flight roll angle

(b) Flight pitch angle

Figure 4.40: TC 3.0 outer-loop control system responses to reference changes. Key: dotted line-reference; solid line- measured output

4.7 Summary

In this chapter, an auto-tuner for cascade PID control system is presented, which includes four steps: (1) identification of the critical frequency information of the closed-loop feedback control system using frequency sampling filter model, (2) calculation of the open-loop frequency response, (3) obtaining an approximate integrator plus delay model, (4) application of PID controller tuning rules.

As it has been discussed in Chapter 3, the commonly used integrator model actually over-simplifies the quadrotor dynamics, which leads to the degraded PID controller performance. Using a high order transfer function in model-based design can complicate the design process, and the resultant PID controller may still not perform as expected due to model-mismatching. By contrast, the key idea behind the proposed auto-tuner is fitting the dominant dynamics of a physical plant into a simplified integrator plus delay model, which is then used to design the PID controller. As a result, the proposed auto-tuner ensures a sound closed-loop control performance without using a complicated high order model. Additionally, as the auto-tuner is fully automatic, it simplifies the PID control tuning process and in the meantime it provides control engineers with reliable results. It has been verified by numerical simulations that, in the auto-tuner's model identification stage, the dominant dynamics of physical systems with different levels of complexity can generally be fitted into the integrator plus delay model. Another group of simulations shows that even though this auto-tuning method relies on the first order plus delay model to calculate PID parameters, it also works for high order integrating systems, because of the dominating effect of the integrator in those plants. Experimental results achieved on TC1.0 and TC3.0 show that the auto-tuning method can help to achieve the best control performance on a specific hardware. Gain margin and phase margin of TC3.0 platform with PI, PD and PID controllers are analyzed, by using the Nyquist plot. For both the quadrotor outer-loop and inner-loop system, PD controller offers the largest gain margin and phase margin. As a result, PD controller is a better option for the inner-loop system. PID controller should be used for the outer-loop to eliminate steady-state error while remaining a relatively large gain and phase margin.

Chapter 5

Quadrotor Performance Evaluation via Step Response Identification

5.1 Introduction

Even though in the previous chapter a PI controller auto-tuner has been developed, control engineers may still need to know the time domain features of the closed-loop systems. In order to assess the closed-loop control performance of a quadrotor, it is well understood that a dynamic response test is required to reveal whether the closed-loop system has a suitable settling time, under-damped modes, and steady-state error. However, the commonly used numerical dynamic response simulation and direct step response experiments have their own drawbacks. This chapter proposes a novel approach to assess the closed-loop control performance of a quadrotor UAV.

Section 5.2 describes the data acquisition using the relay feedback experiment and step response identification using the frequency-sampling filter (FSF). Section 5.3 presents 3-axis experimental results of the quadrotor step response identification. Section 5.4 compares the proposed identification method, the numerical simulation method and the direct step response experimental test. Section 5.5 concludes this chapter.

5.2 Step Response Identification Using Frequency Sampling Filters

5.2.1 Data acquisition using relay feedback

A standard relay experiment produces in most cases a limit cycle dominated by a single frequency. However, this information may not be sufficient for the estimation of a step response model because it requires more frequency response information to ensure satisfactory estimation results. The strategy we adopt in the identification experiment design was introduced in [Wang et al., 1999] and applied by Wang and Gawthrop [2001] to simulation studies of continuous time system identification and by Wang et al. [2004] to estimation of food extruder model. In this type of experiments, we

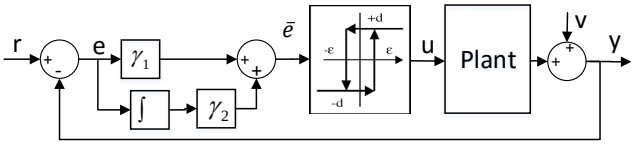


Figure 5.1: Data acquisition: multi-frequency relay feedback system.

make use of multiple relay experiments to generate frequency response information at several frequencies. The proposed apparatus as shown in Figure 5.1 combines in parallel a relay element with that connected in series to an integrator. The experiment is performed by alternating the error signal between the relay path and the integrator-relay path. The design of the experiment then reduces to the selection of this triggering sequence. One of the main benefits of the apparatus is that the design of an identification experiment suitable for obtaining a mathematical model has now been automated. In addition, choice of sampling rate can be set to near continuous measurement.

The relay's input signal $\bar{e}$ is calculated by the following equation:

$$\bar{e(t)} = \gamma_1 e(t) + \gamma_2 \int_0^t e(t)d\tau \tag{5.1}$$
$$\gamma_1 + \gamma_2 = 1,$$

where γ_1 and γ_2 can be either 0 or 1. When $\gamma_1 = 1$ and $\gamma_2 = 0$, the system oscillates at a high frequency and when $\gamma_1 = 0$ and $\gamma_2 = 1$, the system oscillates lower frequencies.

By choosing a set of different γ_1 and γ_2 values, various frequency contents of the input signal can be obtained. In this paper, the γ_1 and γ_2 values are chosen as:

$$\gamma_1 = [1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0]$$
$$\gamma_2 = [0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1].$$

In the quadrotor inner-loop data acquisition relay feedback experiment, the process to be identified is the quadrotor's inner-loop system with proportional control. The process input signal u is generated by the relay, which is also the angular velocity set point to the inner-loop control system. The process output signal y is the quadrotor angular velocity. In the quadrotor outer-loop data acquisition relay feedback experiment, the process to be identified is the quadrotor's outer closed-loop control system. The process input signal u is the signal generated by the relay component, which is also used as the reference signal for the Euler angle. The process output signal y is the measured Euler angle.

Since the process output is corrupted with noise, hysteresis is added to the relay to reduce the effect of the noise by preventing the random switches due to noise, where it produces a dead-zone with its size defined by the parameter ϵ. It is well known that if the width of the hysteresis ϵ equals zero, then the oscillation frequency corresponds to the cross over frequency of the process under the feedback control [Åström and Hägglund, 1984]. An integrator in series to the relay element generates a stable oscillation with the dominant frequency corresponding to $-90°$ on the Nyquist plot [Åström and Hägglund, 1984].

5.2.2 Step response identification using frequency sampling filters

Assume that the continuous time system is stable , the system is sampled uniformly with an interval Δt, and the system has a settling time T_s such that when $t \geq T_s$, the impulse response $h(t) \approx 0$. The corresponding discrete parameter to T_s is $N_s = \frac{T_s}{\Delta t}$. where$N_s$ is the number of samples within the settling time, n is an odd number to represent the number of frequencies included in the Frequency Sampling Filters model; ω_{ds} is the fundamental sampling frequency defined by $\omega_{ds} = \frac{2\pi}{N_s}$. Suppose that $u(k)$ is the process input, $y(k)$ is the process output and $v(k)$ is the disturbance signal. The

output $y(k)$ can be expressed in a linear regression form by defining the parameter vector and the regressor vector as

$$\boldsymbol{\theta} = [T(e^{j0})\ T(e^{j\omega_{ds}})\ T(e^{-j\omega_{ds}})...T(e^{j\frac{n-1}{2}\omega_{ds}})\ T(e^{-j\frac{n-1}{2}\omega_{ds}})]^{T*}$$

and the regressor vector as

$$\boldsymbol{\Phi}(k) = [f(k)^0\ f(k)^1\ f(k)^{-1}...\ f(k)^{\frac{n-1}{2}}\ f(k)^{-\frac{n-1}{2}}]^{T*}$$

where

$$f(k)^l = \frac{1}{N}\frac{1-z^{-N}}{1-e^{jl\omega_{ds}z^{-1}}}$$

for $l = 0, \pm1, \pm2, ..., \pm\frac{n-1}{2}$ and $\boldsymbol{A}^*$ is defined as complex conjugate transpose of $\boldsymbol{A}$. This allows us express the output signal as

$$y(k) = \boldsymbol{\Phi}(k)^*\boldsymbol{\theta} + \boldsymbol{v}(k) \tag{5.2}$$

Figure 5.2 shows the system input-output relationship using frequency-sampling filter model. Given a set of sampled finite amount of data $\{y(1), y(2), y(3), ...\}$ and

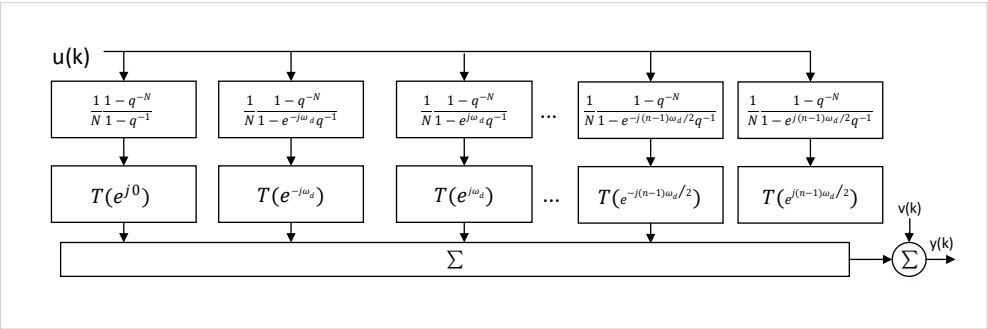


Figure 5.2: Block diagram of frequency-sampling filter structure.

$\{u(1), u(2), u(3), ...\}$, we can obtain an estimate of the frequency-sampling filter model using the same standard recursive least squares algorithm mentioned in chapter 4.

In order to obtain the estimated step response from the estimated frequency parameter vector $\boldsymbol{\theta}$, it can be easily verified [Wang and Cluett, 2000] that the step response of the system at the sample m is in a linear relation to the frequency parameter vector

$\boldsymbol{\theta}$ via

$$g_m = \boldsymbol{Q}(m)^T \boldsymbol{\theta} \tag{5.3}$$

where

$$\boldsymbol{Q}(m) = \begin{bmatrix} \frac{m+1}{N} \\ 2Re(S(1,m)) \\ 2Im(S(1,m)) \\ \vdots \\ 2Re(S(\frac{n-1}{2},m)) \\ 2Im(S(\frac{n-1}{2},m)) \end{bmatrix}$$

$S(l,m) = \frac{1}{N}\frac{1-e^{jl\omega_{ds}(m+1)}}{1-e^{jl\omega_{ds}}}$, $l = 1, 2, \ldots, \frac{n-1}{2}$.

5.3 Quadrotor TC1.0 step Response Identification Experimental Results

5.3.1 Quadrotor TC1.0 Inner-loop Step Response Identification

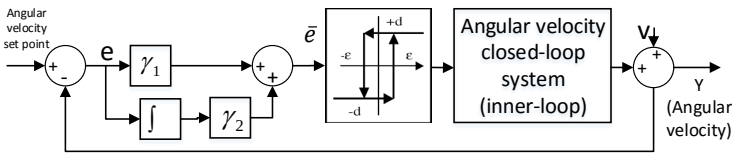
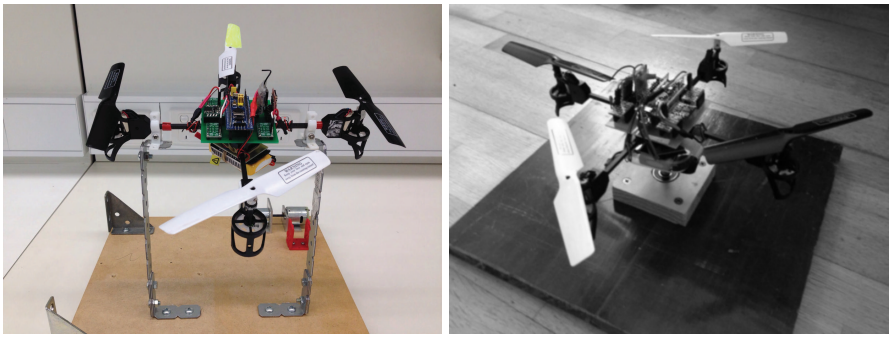


Figure 5.3: Quadrotor TC1.0 inner-loop data acquisition relay feedback system

The relay feedback system used for quadrotor inner-loop data acquisition is shown in Figure 5.3. In the inner-loop relay feedback experiments, the relay device is set to switch between -0.8 and 0.8. The hysteresis is chosen to be $\epsilon = 0.2$ to avoid possible random switches due to the existence of measurement noise. The sampling interval Δt is $0.01s$ for the experiments.

The relay apparatus for x and y axis identification is shown in Figure 5.4(a), which is the same as the one used in the auto-tuner. The z axis relay apparatus is shown in Figure 5.4(b). Figure 5.5 shows the input signals generated by the relay apparatus and responses of 3-axis inner-loop systems. The estimated quadrotor inner-loop step

responses using the frequency sampling filter model are shown in Figure 5.6 with statistical confidence bounds. It is seen from this figure that the $x-$axis ($\dot{\phi}(t)$) and $y-$axis ($\dot{\theta}(t)$) inner closed-loop responses are oscillating, but without steady-state error. The $z-$axis response ($\dot{\phi}(t)$) is slightly faster with a smaller over-shoot. This is because the $x-$axis and $y-$axis inner-loop control systems have a high gain feedback control, which leads to the oscillation. Because all the inner-loop systems have an integrator in the plant dynamics, the closed-loop responses have no steady-state error to reference response. The statistical confidence bounds also indicate a high signal to noise ratio in the experiment.

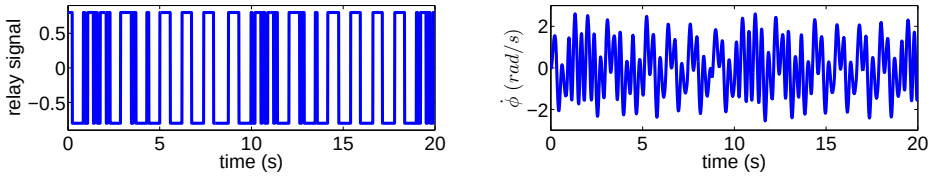


(a) Roll and pitch stand (b) Yaw stand

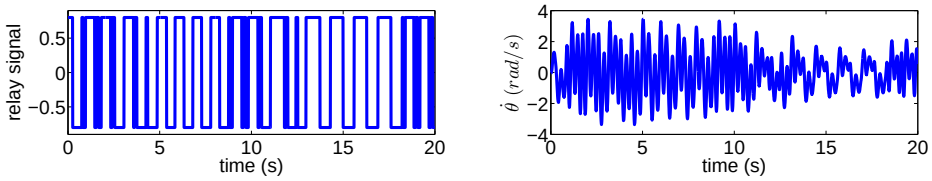
Figure 5.4: Quadrotor TC1.0 testbed

The estimated inner-loop output responses are compared with the experimental responses, shown in Figure 5.7. The estimated output responses are generated by using the same relay input signals as the inputs to the estimated inner-loop frequency sampling filter models. Again, it is seen that the estimated responses are almost identical with the experimental responses.

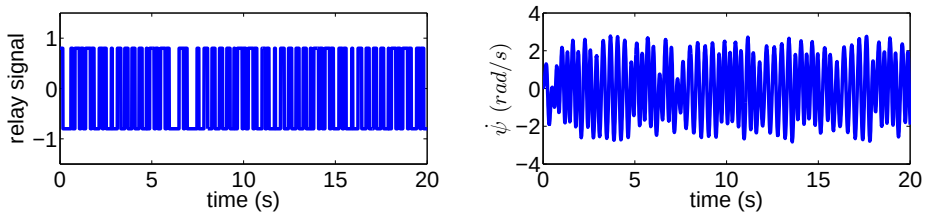
Figure 5.8 shows the estimated inner-loop step responses with different controller settings where the proportional controller gain is increased from $K_p = 0.03$ to 0.11. It can be seen that, with higher controller gain, the closed-loop step response gradually becomes more oscillatory. Increasing the controller gain even further would lead to instability of the closed-loop control system. This set of estimated step responses reveals the closed-loop control system performance with respect to the selection of the proportional controller K_p.



(a) Roll angler velocity relay experimental data



(b) Pitch angler velocity relay experimental data



(c) Yaw angler velocity relay experimental data

Figure 5.5: Quadrotor TC1.0 inner-loop step response identification experimental data.

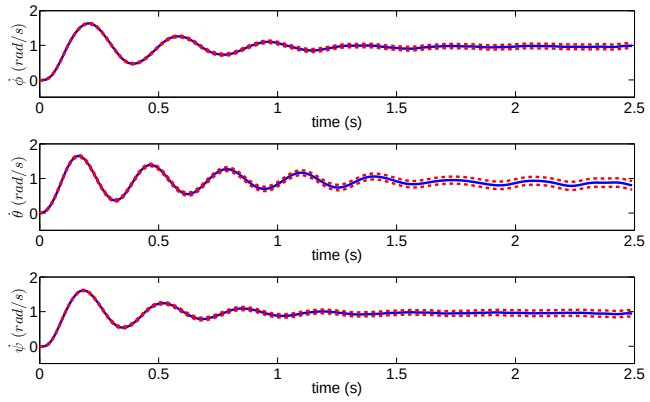


Figure 5.6: Estimated quadrotor TC1.0 inner-loop step responses. Key: solid lines step response; dotted lines 2δ confidence bounds

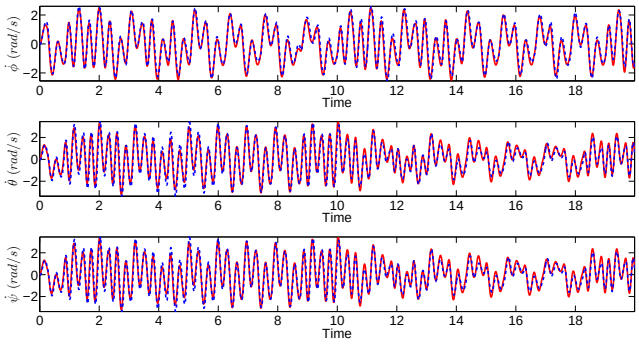
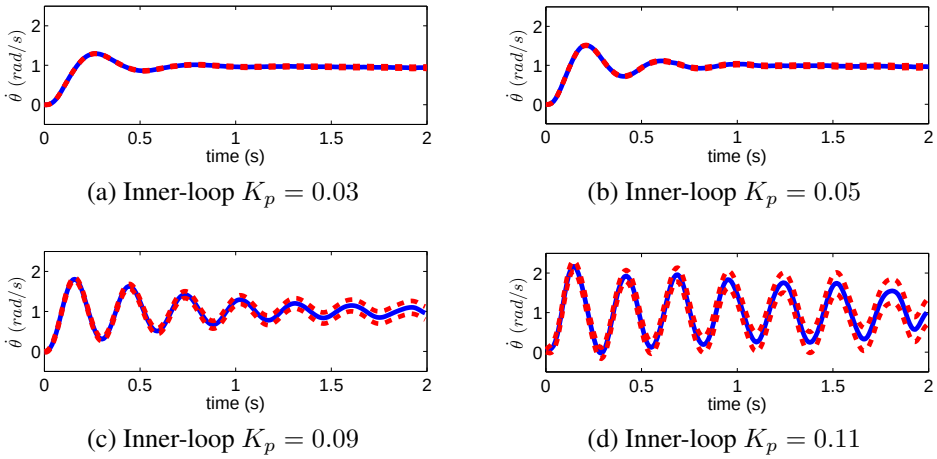


Figure 5.7: Actual and estimated quadrotor TC1.0 inner-loop responses under the same relay signals. Key: solid lines estimated responses; dotted lines actual responses



(a) Inner-loop $K_p = 0.03$

(b) Inner-loop $K_p = 0.05$

(c) Inner-loop $K_p = 0.09$

(d) Inner-loop $K_p = 0.11$

Figure 5.8: Estimated quadrotor TC1.0 inner-loop step responses with different controller gains. Key: solid lines step response; dotted lines 2δ confidence bounds

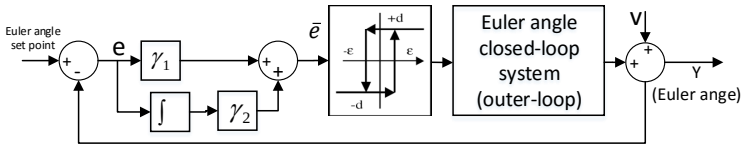
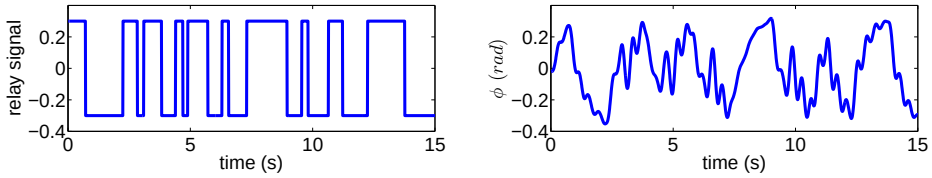


Figure 5.9: Quadrotor TC1.0 outer-loop data acquisition relay feedback system

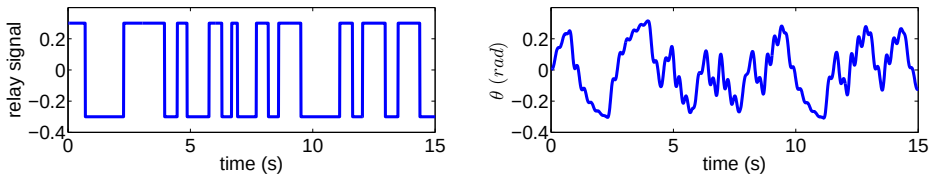
5.3.2 Quadrotor TC1.0 Outer-loop Step Response Estimation

The relay feedback system used for quadrotor outer-loop data acquisition is shown in Figure 5.9. Figure 5.10 shows the input signals generated by the relay apparatus and

the responses of roll and pitch angles. In the outer-loop relay feedback control experiments, the relay element is set to switch between -0.3 and 0.3. The hysteresis is chosen to be 0.1 to avoid possible random switches caused by the measurement noise. The sampling interval Δt is $0.01s$. The estimated outer-loop step responses with confidence bounds using frequency sampling filter model are shown in Figure 5.11. From this figure, it is seen that the quadrotor's $x-$axis and $y-$axis outer-loops have almost the identical dynamics, with roughly 2 second settling time. Also, it can seen that the responses are not smooth on the transient dynamics and the closed-loop responses are oscillatory. There is no steady-state error in the outer-loop closed-loop responses. Overall, the control system performance may need improvement on the transient response. Clearly, the characteristics of closed-loop control system can then be precisely determined from the estimated step responses. Additionally, the measurement noise level is quite low as it is evident from the relatively tight statistical confidence bound.



(a) Roll angle experimental data



(b) Pitch angle experimental data

Figure 5.10: Quadrotor outer-loop step response identification experimental data

To confirm the quality of the estimated model, the predicted output responses are compared with the measured output responses, shown in Figure 5.12. Here, the predicted output responses are generated by using the same relay input signals as the input signals to the estimated outer-loop frequency sampling filter models. It is seen that the predicted responses are almost identical with the measured responses.

To further illustrate the importance of performance assessment, Figure 5.13 shows the estimated outer closed-loop step responses with different controller settings with

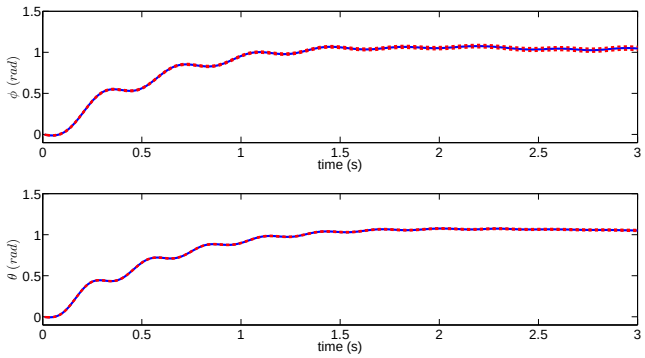


Figure 5.11: Estimated quadrotor outer-loop step responses. Key: solid lines step response; dotted lines 2δ confidence bounds

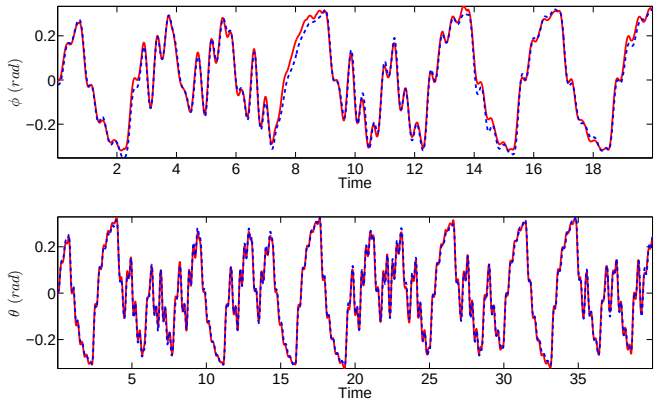
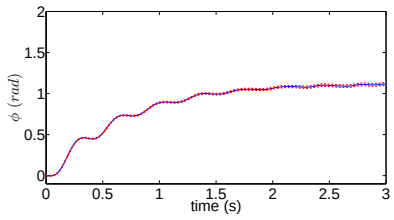


Figure 5.12: Actual and estimated quadrotor outer-loop responses under the same relay signals. Key: solid lines estimated responses; dotted lines actual responses

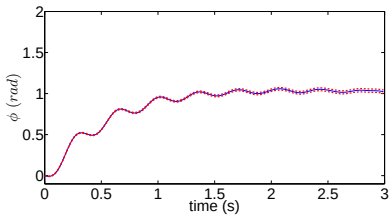
a gradual increase in both proportional and integral control gains. It can be seen that, with higher controller gain, the closed-loop step response speed becomes faster, however it exhibits more oscillatory behavior.

5.3.3 Quadrotor 3-axis Simulation Using Identified Models

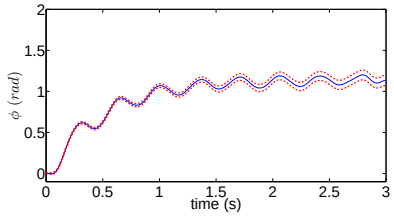
In the previous section, comparisons have been made between the quadrotor experimental responses and the predicted responses and they are almost identical when the quadrotor is fixed on the test-bed. To evaluate the control performance in the realistic



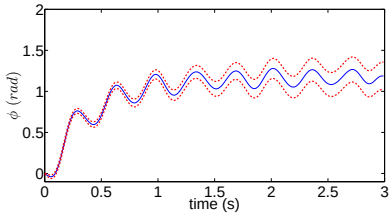
(a) outer-loop $K_p = 1.71$ $K_I = 0.53$



(b) outer-loop $K_p = 1.95$ $K_I = 0.69$



(c) outer-loop $K_p = 2.26$ $K_I = 0.93$



(d) outer-loop $K_p = 2.73$ $K_I = 1.33$

Figure 5.13: Estimated quadrotor inner-loop step responses with different controller gains. Key: solid lines step response; dotted lines 2δ confidence bounds

condition, flying tests are conducted. The predicted responses are compared with the actual responses from a flying test.

The flying test was performed on a quadrotor platform in an indoor environment. This quadrotor platform equipped with 3-axis accelerometers and 3-axis gyroscopes is capable of tracking roll and pitch angle references and yaw angular rate reference. During the flying test, the reference signals ϕ^*, θ^* and $\dot{\psi}^*$ are manually changed using a RC transmitter. These reference signals are then received by the RC receiver on the quadrotor and sent to the on-board micro controller. The micro controller compares the difference between the reference signals from the receiver and the actually feedback signals from the sensors and generates the manipulated variables to stabilize the quadrotor system. The quadrotor's data logger records the flying test data, including reference signals, feedback signals, manipulated variables, sampling instance and the controller parameters.

After the flying test, the recorded reference signals ϕ^*, θ^* and $\dot{\psi}^*$ were fed into the previously identified models to get the estimation of the quadrotor responses under such reference signals. As shown in Figure 5.14, the $x-$axis and $y-$axis predicted outer-loop responses have similar characteristics to the experimental responses, but

not identical. When the quadrotor is flying, there are disturbances due to air flow and those disturbances are not captured by the dynamics models. It is interesting to note that the predicted and actual responses of $z-$axis are almost identical. That is probably because the disturbance acting on $z-$axis during the flying test is smaller than those acting on the $x-$axis and $y-$axis.

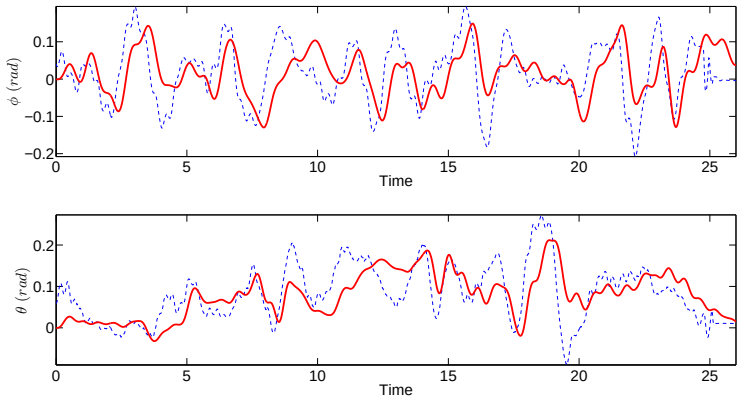


Figure 5.14: Estimated and actual quadrotor responses with the same reference signals. Key: dotted line- measured angles; solid lines predicted angles

5.4 Comparisons of Common Dynamic Response Methods

The numerical simulation and step response experimental tests are two commonly used means to analyse the quadrotor's closed-loop dynamics. In this section, comparisons are made among the step responses obtained from numerical simulations, the direct step responses experiments, and those obtained by using the proposed model identification method.

5.4.1 Comparisons with nonlinear simulations

As the quadrotor physical plant parameters can be measured, the nonlinear model of the quadrotor is available and it is possible to assess the closed-loop control performance by simulation of the closed-loop system. However, if the measured physical

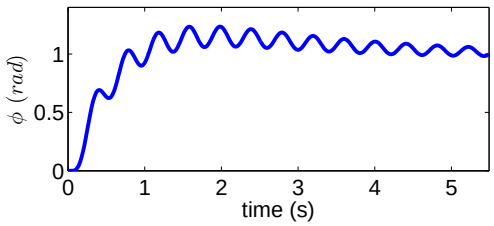
parameters contain error, the simulation results may not reveal the true dynamics of the closed-loop system. In order to show the difference between the step responses achieved from the proposed model identification method and those from numerical simulation, the following comparisons are performed.

Table 5.1: Quadrotor actual and measured physical parameters

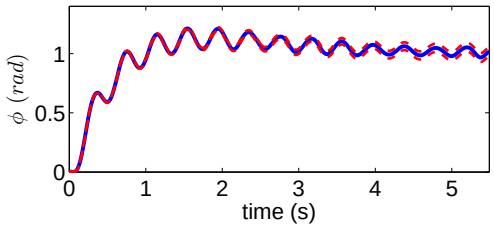
Parameter	**Actual**	**Measured 1**	**error**	**Measured 2**	**error**	**unit**
I_{xx}	3.2143×10^{-4}	3.8572×10^{-4}	$+20\%$	2.8929×10^{-4}	-10%	$kg \cdot m^2$
I_{yy}	3.2143×10^{-4}	3.8572×10^{-4}	$+20\%$	2.8929×10^{-4}	-10%	$kg \cdot m^2$
I_{zz}	4.6087×10^{-4}	5.5304×10^{-4}	$+20\%$	4.1478×10^{-4}	-10%	$kg \cdot m^2$
b_t	9.6756×10^{-7}	7.7405×10^{-7}	-20%	1.0159×10^{-6}	$+20\%$	N/A
r_{wv}	137.6571	123.8914	-10%	144.54	$+5\%$	rad/Vs
k_d	1.8336×10^{-8}	1.6502×10^{-8}	-10%	1.9253×10^{-8}	$+5\%$	N/A
d_m	0.032	0.0256	-20%	0.0336	$+5\%$	s
ϵ_m	0.072	0.0648	-10%	0.0756	$+5\%$	s

In Table 5.1, the first column lists all quadrotor parameters used in the comparisons. As absolute accurate physical values are never available, the values in the second column are 'assumed' to be the true values. And in the third column, some measurement errors are intentionally added to the true values. The forth column shows the errors in percentage. The errors vary from -20% to $+20\%$. Another set of values with errors are given in the fifth column, and the errors in percentage are shown in the sixth column. The errors vary from -10% to $+20\%$.

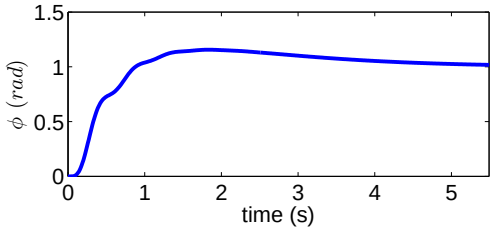
Unit step responses of the quadrotor outer-loop system are given in Figure 5.15. It should be noted that even though all plots in 5.15 are from numerical simulations, they are different in nature. Figure 5.15 (a) is assumed to be the true step response of the quadrotor system as it is achieved by using the true physical parameters shown in the second column of Table 5.1. The step response shown in Figure 5.15 (b) is obtained by using the proposed model identification method. Obviously, it is closely matched to Figure 5.15 (a). In contrast, Figure 5.15(c) is obtained from a simulation by using the first set of parameters with errors. As it can be seen, the step response is quite different from Figure 5.15 (a) with much less oscillation and slightly lower overshoot. When using the second set of physical parameters with errors, the step response even becomes unstable, which is shown in Figure 5.15 (d).



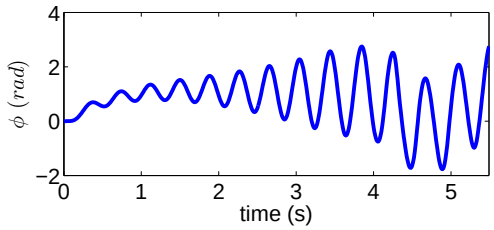
(a) Actual step response



(b) Estimated step response. Key: solid lines step response; dotted lines 2δ confidence bounds



(c) Step response simulation with inaccurate parameters set 1



(d) Step response simulation with inaccurate parameters set 2

Figure 5.15: Quadrotor outer-loop step responses.

5.4.2 Comparisons with step response experiments

The quadrotor TC1.0's direct inner-loop step response experimental test is conducted as shown in Figure 5.17. It can been seen from the figure, the step response is seriously corrupted by disturbances. The step response parameters such as rise time, overshoot, and settling time can hardly be determined. The characteristic of under-damped closed-loop system was masked by the disturbance.

Additionally, a step response without the relay feedback control for the outer-loop system (roll angle) is shown in Figure 5.16. Perhaps, one could say that the outer-loop system's time-domain specifications such as rise time, settling time could be roughly read off from the figure and used to evaluate the designed controller. However, if we carefully compare the positive and the negative step responses in Figure 5.16, it can be seen that the overshoot and settling time are quite different, which are mainly caused by the disturbance on the control signal. As a result, the result obtained from a normal step response test may not be reliable as it is subject to the disturbance level when conducting the test.

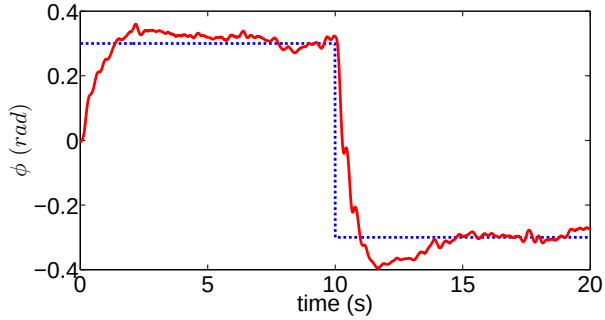


Figure 5.16: Quadrotor TC1.0 outer closed-loop step response. Key: dotted line set point signal; solid line output signal

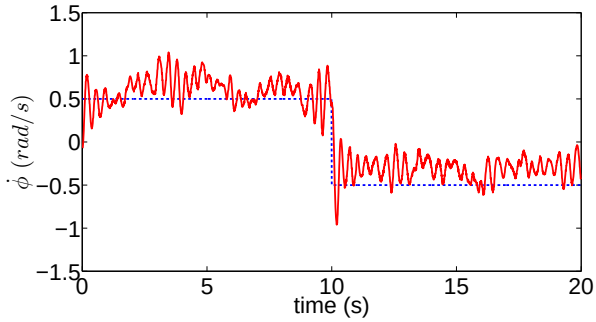


Figure 5.17: Quadrotor TC1.0 inner-loop step response. Key: dotted line set point signal; solid line output signal

As it has been discussed above, numerical simulations and step response experiments may not provide satisfactory results to help control engineers understand the true dynamics of the quadrotor closed-loop system and it thus becomes the motivation of the proposed model identification method.

5.5 Conclusions

This chapter presents an approach to identify the quadrotor's 3-axis closed-loop step responses for controller performance evaluation. The first step in the proposed method is data acquisition using relay feedback experiment and the second step is step response identification using frequency sampling filters with recursive least squares method.

The proposed step response identification method is applied to quadrotor TC1.0's

inner-loop and outer-loop systems. The identification results show very clean step response curves, which can be used to accurately determine the closed-loop systems' time domain features. To validate the proposed identification method, the reference signals recorded in the relay experiments are fed into the identified models. The identified models then generate very similar outputs as what is recorded in the relay tests. Additionally, the reference signals generated in the flying tests are also fed into the identified model. The identified models' outputs are close to the flying test results, but errors exist. It should be emphasized here that the proposed model identification method is not to replace the actual flying test. Its advantages over the numerical simulation and the normal step response test make it a more reliable way to evaluate the designed closed-loop system performance in a safe environment. With the implication of the performance assessment on ground, early fault detection for quadrotor UAVs could be performed.

Comparisons are made between the proposed identification method and numerical simulations. As it has been shown, the step response achieved by using the identification method is very close to the true step response, while numerical simulations give inaccurate results due to uncertainty of plant physical parameters. It thus comes to the conclusion that in the circumstance that the plant parameters are likely to be inaccurate, the proposed identification method will provide better prediction of the system's performance.

Comparisons are also made between the estimated step responses and the direct step response experimental tests conducted on the same test-bed. The results show that the estimated step responses give a clearer and more intuitive presentation which can be directly used by engineers to evaluate and improve the designed controllers, when direct step response experimental results are seriously corrupted by disturbances.

Chapter 6

Model Predictive Control of Quadrotor

6.1 Introduction

In previous chapters, cascade PID controllers have been deployed on the quadrotor system to control its attitude. As the quadrotor is a Multiple-input, multiple-output (MIMO) system, while the PID control is a Single-input, single-output (SISO) control technique, the quadrotor system needs to be decoupled before the controller design. Furthermore, because of the coupling on the control signals, constraints cannot be directly applied on the actual control signals to get the optimized solutions. To solve these problems, model predictive control (MPC) is employed to control the quadrotor's attitude and position. Two MPC control architectures, namely cascade MPC and centralized MPC are compared. Due to the lack of position sensor on the quadrotor platform, numerical simulations are conducted on the quadrotor mathematical model to validate the proposed control algorithm.

This chapter is organized as follows. A discrete-time MPC algorithm based on Laugerre functions is introduced first. Hildreth's Quadratic Programming (QP) algorithm is employed to deal with constraints. The constrained MPC algorithm is then applied to the quadrotor position control problem. Centralized MPC control and cascade MPC control structures are tested and compared in a quadrotor hovering scenario simulation. There are two methods to deal with the constraints on the control signals.

One is to use pure saturation to limit the control signals within a range (e.g. PID control). In comparison, the other one is to find the optimal solutions under constraints (e.g. MPC). To analyze how constraints or actuator saturation affect the control performance, several simulation studies of cascaded MPC quadrotor position control are perform at the end of this chapter.

6.2 Model Predictive Control

6.2.1 Discrete-time Model Predictive Control

For completeness of the thesis, the mathematic expressions and derivations of Laguerre MPC algorithm are briefly explained. For more details of Laguerre MPC, readers are referred to [Wang, 2009].

The plant to be controlled is described by the discrete time model of the form as

$$\begin{aligned} \boldsymbol{x}_m(k+1) &= \boldsymbol{A}_m\boldsymbol{x}_m(k) + \boldsymbol{B}_m\boldsymbol{u}(k) \\ \boldsymbol{y}(k) &= \boldsymbol{C}_m\boldsymbol{x}_m(k). \end{aligned} \tag{6.1}$$

Assume that the plant has p inputs, q outputs and n states. The difference of the state variable is written as

$$\Delta\boldsymbol{x}_m(k+1) = \boldsymbol{x}_m(k+1) - \boldsymbol{x}_m(k), \tag{6.2}$$

and the difference of the control variable by

$$\Delta\boldsymbol{u}(k) = \boldsymbol{u}(k) - \boldsymbol{u}(k-1). \tag{6.3}$$

Then we augment the original state space model (6.1) as

$$\overbrace{\begin{bmatrix}\Delta \boldsymbol{x}_m(k+1)\\ \boldsymbol{y}(k+1)\end{bmatrix}}^{\boldsymbol{x}(k+1)} = \overbrace{\begin{bmatrix}\boldsymbol{A}_m & \boldsymbol{o}_m^T\\ \boldsymbol{C}_m\boldsymbol{A}_m & \boldsymbol{I}_{q\times q}\end{bmatrix}}^{\boldsymbol{A}}\overbrace{\begin{bmatrix}\Delta \boldsymbol{x}_m(k)\\ \boldsymbol{y}(k)\end{bmatrix}}^{\boldsymbol{x}(k)} + \overbrace{\begin{bmatrix}\boldsymbol{B}_m\\ \boldsymbol{C}_m\boldsymbol{B}_m\end{bmatrix}}^{\boldsymbol{B}}\Delta \boldsymbol{u}(k)$$
$$\boldsymbol{y}(k) = \overbrace{\begin{bmatrix}\boldsymbol{o}_m & \boldsymbol{I}_{q\times q}\end{bmatrix}}^{\boldsymbol{C}}\begin{bmatrix}\Delta \boldsymbol{x}_m(k)\\ \boldsymbol{y}(k)\end{bmatrix}, \tag{6.4}$$

where $\boldsymbol{I}_{q\times q}$ is the identity matrix with dimensions $q\times q$, which is the number of outputs; and $\boldsymbol{o}_m$ is a $q \times n_1$ zero matrix. In (6.4), $\boldsymbol{A}_m$, $\boldsymbol{B}_m$ and $\boldsymbol{C}_m$ have dimension $n_1 \times n_1$, $n_1 \times m$ and $q \times n_1$, respectively. For notational simplicity, we denote (6.4) by

$$\begin{aligned}\boldsymbol{x}(k+1) &= \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k)\\ \boldsymbol{y}(k+1) &= \boldsymbol{C}\boldsymbol{x}(k).\end{aligned} \tag{6.5}$$

The triplet $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$ is called the augmented model, which will be used in the design of predictive control. In the following, the dimensionality of the augmented state-space equation is taken to be $n = n_1 + q$. Let us denote the vector

$$\boldsymbol{Y} = \begin{bmatrix}\boldsymbol{y}(k+1)\\ \boldsymbol{y}(k+2)\\ \boldsymbol{y}(k+3)\\ \vdots\\ \boldsymbol{y}(k+N_p)\end{bmatrix};\ \Delta\boldsymbol{U} = \begin{bmatrix}\Delta\boldsymbol{u}(k)\\ \Delta\boldsymbol{u}(k+1)\\ \Delta\boldsymbol{u}(k+2)\\ \vdots\\ \Delta\boldsymbol{u}(k+N_c-1)\end{bmatrix}. \tag{6.6}$$

Effectively, we have

$$\boldsymbol{Y} = \boldsymbol{F}_M\boldsymbol{x}(k) + \boldsymbol{\Phi}_M\Delta\boldsymbol{U}, \tag{6.7}$$

where

$$\boldsymbol{F}_M = \begin{bmatrix}\boldsymbol{C}\boldsymbol{A} & \boldsymbol{C}\boldsymbol{A}^2 & \boldsymbol{C}\boldsymbol{A}^3 & \dots & \boldsymbol{C}\boldsymbol{A}^{N_p}\end{bmatrix}^T,$$

$$\boldsymbol{\Phi}_M = \begin{bmatrix} \boldsymbol{CB} & 0 & 0 & ... & 0 \\ \boldsymbol{CAB} & \boldsymbol{CB} & 0 & ... & 0 \\ \boldsymbol{CA^2B} & \boldsymbol{CAB} & \boldsymbol{CB} & ... & 0 \\ \vdots & \vdots & \vdots & \vdots & \\ \boldsymbol{CA^{N_p-1}B} & \boldsymbol{CA^{N_p-2}B} & \boldsymbol{CA^{N_p-3}B} & ... & \boldsymbol{CA^{N_p-N_c}B} \end{bmatrix}.$$

$\boldsymbol{I}_{q\times q}$ is a $q \times q$ identity matrix, $\boldsymbol{R}_s$ is the set point vector, $\overline{\boldsymbol{R}}$ is the weighting matrix for control signal, N_c is control horizon, N_p is prediction horizon.

The cost function of traditional MPC is expressed as

$$J = (\boldsymbol{R}_s - \boldsymbol{F}_M\boldsymbol{x}(k))^T(\boldsymbol{R}_s - \boldsymbol{F}_M\boldsymbol{x}(k)) - 2\Delta\boldsymbol{U}^T\boldsymbol{\Phi}_M^T(\boldsymbol{R}_s - \boldsymbol{F}_M\boldsymbol{x}(k)+ \\ \Delta\boldsymbol{U}^T(\boldsymbol{\Phi}_M^T\boldsymbol{\Phi}_M + \overline{\boldsymbol{R}})\Delta\boldsymbol{U}. \quad (6.8)$$

Now we calculate derivation of the cost function J:

$$\frac{dJ}{d\Delta\boldsymbol{U}} = -2\boldsymbol{\Phi}_M^T(\boldsymbol{R}_s - \boldsymbol{F}_M\boldsymbol{x}(k)) + 2(\boldsymbol{\Phi}_M^T\boldsymbol{\Phi}_M + \overline{\boldsymbol{R}})\Delta\boldsymbol{U}. \quad (6.9)$$

The necessary condition of the minimum J is obtained:

$$\frac{dJ}{d\Delta\boldsymbol{U}} = 0. \quad (6.10)$$

The optimal solution for the control signal is

$$\Delta\boldsymbol{U} = (\boldsymbol{\Phi}_M^T\boldsymbol{\Phi}_M + \overline{\boldsymbol{R}})^{-1}(\boldsymbol{\Phi}_M^T\overline{\boldsymbol{R}}_s\boldsymbol{r}(k) - \boldsymbol{\Phi}_M^T\boldsymbol{F}_M\boldsymbol{x}(k)), \quad (6.11)$$

where matrix $\boldsymbol{\Phi}_M^T\boldsymbol{\Phi}_M$ has dimension $mN_c \times mN_c$ and $\boldsymbol{\Phi}_M^T\boldsymbol{F}_M$ has dimension $mN_c \times n$ and $\boldsymbol{\Phi}_M^T\overline{\boldsymbol{R}}_s$ equals the last q columns of $\boldsymbol{\Phi}_M^T\boldsymbol{F}_M$. The weight matrix $\overline{\boldsymbol{R}}$ is a block matrix with m blocks and has its dimension equal to dimension of $\boldsymbol{\Phi}_M^T\boldsymbol{\Phi}_M$. The set-point signal is $\boldsymbol{r}(k) = \begin{bmatrix} \boldsymbol{r}_1(k) & \boldsymbol{r}_2(k) & ... & \boldsymbol{r}_q(k) \end{bmatrix}^T$ as the q set-point signals to the multi-output system.

Applying the receding horizon control principle, the first m elements in $\Delta\boldsymbol{U}$ are

taken to form the incremental optimal control:

$$\begin{aligned}\Delta \boldsymbol{u}(k) = \begin{bmatrix} \boldsymbol{I}_m & \boldsymbol{o}_m & \dots & \boldsymbol{o}_m \end{bmatrix} (\boldsymbol{\Phi}_M^T \boldsymbol{\Phi}_M + \overline{\boldsymbol{R}})^{-1} (\boldsymbol{\Phi}_M^T \overline{\boldsymbol{R}}_s \boldsymbol{r}(k) - \boldsymbol{\Phi}_M^T \boldsymbol{F}_M \boldsymbol{x}(k)) \\ = \boldsymbol{K}_{\boldsymbol{y}} \boldsymbol{r}(k) - \boldsymbol{K}_{mpc} \boldsymbol{x}(k)\end{aligned} \tag{6.12}$$

Note that the characteristic polynomial equation of the augmented model is

$$det \begin{bmatrix} \lambda \boldsymbol{I} - \boldsymbol{A}_m & \boldsymbol{o}_m^T \\ -\boldsymbol{C}_m \boldsymbol{A}_m & (\lambda - 1)\boldsymbol{I}_{q \times q} \end{bmatrix} = (\lambda - 1)^q det(\lambda \boldsymbol{I} - \boldsymbol{A}_m) = 0, \tag{6.13}$$

where we use the property that the determinant of a block lower triangular matrix equals the product of the determinants of the matrices on the diagonal. Hence, the eigenvalues of the augmented model are the union of the eigenvalues of the plant model and the q eigenvalues, $\lambda = 1$. This means that there are q integrators embedded into the augmented design model. This is the means that we use to obtain integral action for the MPC systems.

6.2.2 Discrete-time MPC Using Laguerre Functions

Now we model the control signal with Laguerre functions. Laguerre functions are a set of discrete orthonormal basis functions that can be expressed as

$$\begin{aligned}\Gamma_N(z) &= \Gamma_{N-1}(z)(\frac{z^{-1} - a}{1 - az^{-1}}) \\ \Gamma_1(z) &= \frac{\sqrt{1 - a^2}}{1 - az^{-1}}.\end{aligned} \tag{6.14}$$

Letting $l_i(k)$ denote the inverse z-transform of $\Gamma_i(z, a)$

$$\boldsymbol{L}(k) = \begin{bmatrix} l_1(k) & l_2(k) & \dots & l_N(k) \end{bmatrix}^T,$$

the set of discrete-time Laguerre functions satisfies the following difference equation:

$$\boldsymbol{L}(k+1) = \boldsymbol{A}_1 \boldsymbol{L}(k), \tag{6.15}$$

where the initial condition is given by

$$\boldsymbol{L}(0)^T = \sqrt{\beta}\begin{bmatrix}1 & -a & a^2 & -a^3 & ... & (-1)^{N-1}a^{N-1}\end{bmatrix},$$

a is the Laguerre scaling factor, $\beta = (1-a^2)$, and A_1 is a $N \times N$ matrix. For example $N = 5$,

$$\boldsymbol{A}_1 = \begin{bmatrix} a & 0 & 0 & 0 & 0 & 0 \\ \beta & a & 0 & 0 & 0 & \\ -a\beta & \beta & a & 0 & 0 & \\ a^2\beta & -a\beta & \beta & a & 0 & \\ -a^3\beta & a^2\beta & -a\beta & \beta & a & \end{bmatrix},$$

$$\boldsymbol{L}(0)^T = \sqrt{\beta}\begin{bmatrix}1 & -a & a^2 & -a^3 & a^4\end{bmatrix}^T.$$

Letting

$$\Delta\boldsymbol{u}(k) = \begin{bmatrix}\Delta u_1(k) & \Delta u_2(k)...\Delta u_m(k)\end{bmatrix}^T$$

, the input matrix can be partitioned to $\boldsymbol{B} = \begin{bmatrix}\boldsymbol{B}_1 & \boldsymbol{B}_2 & ... & \boldsymbol{B}_p\end{bmatrix}$.

we express the ith control signal $\Delta u_i(k)$ by choosing a scaling factor a_i and order N_i where a_i and N_i are selected for this particular input, such that

$$\Delta u_i(k) = \boldsymbol{L}_i(k)^T\boldsymbol{\eta}_i, \tag{6.16}$$

where $\boldsymbol{\eta}_i$ and $\boldsymbol{L}_i(k)$ are the Laguerre network description of the ith control, specifically

$$\boldsymbol{L}_i(k)^T = \begin{bmatrix}l_1^i(k) & l_2^i(k) & ...l_{N_i}^i(k)\end{bmatrix}. \tag{6.17}$$

Based on the partition of the input matrix and given state variable information at $\boldsymbol{x}(k)$, the prediction of the future state at time m is written as

$$\begin{aligned}\boldsymbol{x}(k+m) &= \boldsymbol{A}^m\boldsymbol{x}(k) + \sum_{j=0}^{m-1}\boldsymbol{A}^{m-j-1}\begin{bmatrix}\boldsymbol{B}_1\boldsymbol{L}_1(J)^T\boldsymbol{B}_2\boldsymbol{L}_2(J)^T...\boldsymbol{B}_m\boldsymbol{L}_m(J)^T\end{bmatrix}\boldsymbol{\eta} \\ &= \boldsymbol{A}^m\boldsymbol{x}(k_i) + \boldsymbol{\Phi}(m)^T\boldsymbol{\eta}, \end{aligned}\tag{6.18}$$

where the parameter vector $\boldsymbol{\eta}$ and the data matrix $\boldsymbol{\phi}(m)^T$ consist of the individual coefficient vectors given by

$$\boldsymbol{\eta}^T = \begin{bmatrix} \boldsymbol{\eta}_1^T & \boldsymbol{\eta}_2^T & ... & \boldsymbol{\eta}_m^T \end{bmatrix}$$
$$\boldsymbol{\phi}(m) = \begin{bmatrix} \boldsymbol{B}_1\boldsymbol{L}_1(J)^T\boldsymbol{B}_2\boldsymbol{L}_2(J)^T...\boldsymbol{B}_m\boldsymbol{L}_m(J)^T \end{bmatrix}.$$

The cost function is defined as

$$J = \boldsymbol{\eta}^T\boldsymbol{\Omega}\boldsymbol{\eta} + 2\boldsymbol{\eta}^T\boldsymbol{\Psi}\boldsymbol{x}(k) + \sum_{m=1}^{N_p} \boldsymbol{x}(k)^T(\boldsymbol{A})^m\boldsymbol{Q}\boldsymbol{A}^m\boldsymbol{x}(k_i), \tag{6.19}$$

where the matrices $\boldsymbol{\Omega}$ and $\boldsymbol{\Psi}$ are

$$\boldsymbol{\Omega} = \sum_{m=1}^{N_p} \boldsymbol{\Phi}(m)\boldsymbol{Q}\boldsymbol{\Phi}(\boldsymbol{m})^T + \boldsymbol{R_L};\ \boldsymbol{\Psi} = \sum_{m=1}^{N_p} \boldsymbol{\Phi}(m)\boldsymbol{Q}\boldsymbol{A}^m.$$

Without constraints, the optimal solution of the cost function 6.19 is given by

$$\boldsymbol{\eta} = -\boldsymbol{\Omega}^{-1}\boldsymbol{\Psi}\boldsymbol{x}(k_i).$$

Upon obtaining the optimal parameter vector $\boldsymbol{\eta}$, the receding horizon control law is realised as

$$\Delta\boldsymbol{u}(k) = \begin{bmatrix} \boldsymbol{L}_1(0)^T & \boldsymbol{o}_2^T & ... & \boldsymbol{o}_m^T \\ \boldsymbol{o}_1^T & \boldsymbol{L}_2(0)^T & ... & \boldsymbol{o}_2^T \\ ... & ... & \ddots & \vdots \\ \boldsymbol{o}_1^T & \boldsymbol{o}_2^T & ... & \boldsymbol{L}_m(0)^T \end{bmatrix} \boldsymbol{\eta}, \tag{6.20}$$

where $\boldsymbol{o}_k^T, k = 1, 2, ...m$ represents a zero block row vector with identical dimension to $\boldsymbol{L}_k(0)^T$. The control varialbe $\Delta\boldsymbol{u}(k)$ can be written in the form of linear state feedback control assuming that the future reference trajectories are constant within the prediction horizon. Namely,

$$\Delta\boldsymbol{u}(k) = -\boldsymbol{K}_{mpc}\boldsymbol{x}(k), \tag{6.21}$$

where the state feedback control gain matrix $\boldsymbol{K}_{mpc}$ is

$$\boldsymbol{K}_{mpc} = \begin{bmatrix} \boldsymbol{L}_1(0)^T & \boldsymbol{o}_2^T & ... & \boldsymbol{o}_m^T \\ \boldsymbol{o}_1^T & \boldsymbol{L}_2(0)^T & ... & \boldsymbol{o}_2^T \\ ... & ... & \ddots & \vdots \\ \boldsymbol{o}_1^T & \boldsymbol{o}_2^T & ... & \boldsymbol{L}_m(0)^T \end{bmatrix} \boldsymbol{\Omega}^{-1}\boldsymbol{\Psi}.$$

With the definition of state feedback control gain matrix $\boldsymbol{K}_{mpc}$, the closed-loop feedback control is

$$\boldsymbol{x}(k+1) = (\boldsymbol{A} - \boldsymbol{B}\boldsymbol{K}_{mpc})\boldsymbol{x}(k) \tag{6.22}$$

and closed-loop stability and performance of the predictive control system can be checked by examining the location of its eigenvalues.

6.2.3 Discrete-time MPC with Constraints

The object of model predictive control is to solve an optimization problem that takes into account the constraints. Although model predictive control is able to deal with many kinds of constraints either on control signals or on output signals, here we will only focus on constraints on the amplitude of the control variables. Optimization in MPC is realized by minimizing the object function subject to some constraints, which can be considered as a quadratic programming problem[Wang, 2009].

We suppose the constraints are given for the upper limits as

$$\boldsymbol{u}^{max} = \begin{bmatrix} u_1^{max} & u_2^{max} & ... & u_m^{max} \end{bmatrix}^T,$$

and lower limits as

$$\boldsymbol{u}^{min} = \begin{bmatrix} u_1^{min} & u_2^{min} & ... & u_m^{min} \end{bmatrix}^T.$$

Then each control signal is required to satisfy the constraints:

$$\begin{array}{c} u_1^{min} \leqslant u_1 \leqslant u_1^{max} \\ u_2^{min} \leqslant u_2 \leqslant u_2^{max} \\ \vdots \\ u_m^{min} \leqslant u_3 \leqslant u_m^{max} \end{array}. \tag{6.23}$$

Writing equation (6.23) in the vector form gives:

$$\boldsymbol{u}^{min} \leqslant \boldsymbol{u} \leqslant \boldsymbol{u}^{max}$$

which can be expressed by two inequalities:

$$-\boldsymbol{u} \leqslant -\boldsymbol{u}^{min}$$
$$\boldsymbol{u} \leqslant \boldsymbol{u}^{max}$$

In a matrix form, this becomes

$$\begin{bmatrix} -\boldsymbol{I} \\ \boldsymbol{I} \end{bmatrix} \boldsymbol{u} \leqslant \begin{bmatrix} -\boldsymbol{u}^{min} \\ \boldsymbol{u}^{max} \end{bmatrix} \tag{6.24}$$

We denote (6.24) by

$$\boldsymbol{M}\boldsymbol{u} \leqslant \boldsymbol{\gamma}, \tag{6.25}$$

where $\boldsymbol{M}$ is a matrix reflecting the constraints, with its number of rows equal to the number of constraints and number of columns equal to the number of input signals.

Since the cost function J is a quadratic, and the constraints are linear inequalities, the problem of finding an optimal predictive control becomes one of finding an optimal solution to a standard quadratic programming problem.

To be consistent with the literatures of quadratic programming [Wang, 2009], the decision variable is denoted by $\boldsymbol{x}$. The objective function J and the constraints are expressed as

$$J = \boldsymbol{x}^T \boldsymbol{E}\boldsymbol{x} + 2\boldsymbol{x}^T \boldsymbol{F}$$
$$\boldsymbol{M}\boldsymbol{x} \leq \boldsymbol{\gamma}, \tag{6.26}$$

where $\boldsymbol{E}$, $\boldsymbol{F}$, $\boldsymbol{M}$, and $\boldsymbol{\gamma}$ are compatible matrices and vectors in the quadratic programming problem. Without loss of generality, $\boldsymbol{E}$ is assumed to be symmetric and positive definite.

A simple algorithm called Hildreth's Quadratic Programming was proposed to solve quadratic programming problem [Hildreth, 1957]. The iteration expression of

Hildreth's Quadratic Programming Procedure is given in following equation:

$$\lambda_i^{m+1} = max(0, w_i^{m+1})$$
$$w_i^{m+1} = -\frac{1}{g_{ii}}[k + \sum_{j=1}^{i-1} g_{ij}\lambda_j^{m+1} + \sum_{j=i+1}^{n} g_{ij}\lambda_i^m], \tag{6.27}$$

where m means the mth iteration, the scalar g_{ii} is the $ijth$ element in the matrix $\boldsymbol{G} = \boldsymbol{M}\boldsymbol{E}^{-1}\boldsymbol{M}^T$ and k_i is the ith element in the vector $\boldsymbol{K} = \boldsymbol{\gamma} + \boldsymbol{M}\boldsymbol{E}^{-1}F$, $\boldsymbol{\lambda}$ is a column vector called Lagrange multiplier.

When the iteration is completed, the converged Lagrange multiplier $\boldsymbol{\lambda}^\star$ contains either zero or positive values. The constrained minimization over $\boldsymbol{x}$ is given by

$$\boldsymbol{x} = -\boldsymbol{E}^{-1}(\boldsymbol{F} + \boldsymbol{M}^T\boldsymbol{\lambda}^*). \tag{6.28}$$

The constrained model predictive control requires real-time optimization using quadratic programming. Assuming that state-variable information $\boldsymbol{x}(k)$ at the sampling time k is given, and that the lower and upper limits on $\boldsymbol{u}$ are $\boldsymbol{u}^{min}$ and $\boldsymbol{u}^{max}$, the optimization procedure is to minimize the cost function J where

$$J = \boldsymbol{\eta}^T\boldsymbol{\Omega}\boldsymbol{\eta} + 2\boldsymbol{\eta}^T\boldsymbol{\psi}\boldsymbol{x}(k), \tag{6.29}$$

while ensuring that

$$\boldsymbol{u}^{min} \leqslant \boldsymbol{u}(k+m) \leqslant \boldsymbol{u}^{max} \tag{6.30}$$

with $m = 0, 1, 2, ...$

Noting that the increment of the control signal is $\boldsymbol{u}(k) = \sum_{i=0}^{k-1}\Delta\boldsymbol{u}(i)$, then the inequality becomes

$$\boldsymbol{u}^{min} \leqslant \begin{bmatrix} \sum_{i=0}^{k-1}\boldsymbol{L}_1(i)^T & \boldsymbol{o}_2^T & ... & \boldsymbol{o}_m^T \\ \boldsymbol{o}_1^T & \sum_{i=0}^{k-1}\boldsymbol{L}_2(i)^T & ... & \boldsymbol{o}_2^T \\ ... & ... & \ddots & \vdots \\ \boldsymbol{o}_1^T & \boldsymbol{o}_2^T & ... & \sum_{i=0}^{k-1}\boldsymbol{L}_m(i)^T \end{bmatrix} \leqslant \Delta\boldsymbol{u}^{max}, \tag{6.31}$$

where $\boldsymbol{u}(k_1 - 1)$ is the previous control signal, and $\boldsymbol{o}_k^T$ is a zero row vector with the same dimension as $\boldsymbol{L}_k(0)^T$.

6.3 Quadrotor TC2.0 Centralized MPC Control

A centralized MPC controller is applied to the quadrotor to control its positions and yaw angle subject to control signal constraints. The centralized MPC control block diagram is given in Figure 6.1. The state variable is defined as

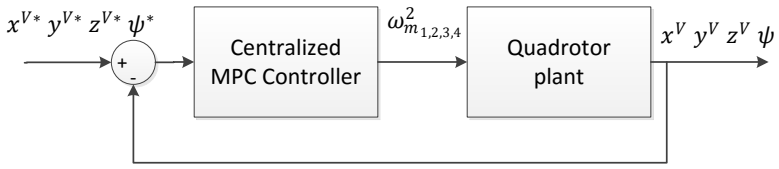


Figure 6.1: Quadrotor centralized MPC control diagram

$$\boldsymbol{x}_m = \begin{bmatrix} x^V\ y^V\ z^V\ \dot{x}^V\ \dot{y}^V\ \dot{z}^V\ \phi\ \theta\ \psi\ \dot{\phi}\ \dot{\theta}\ \dot{\psi} \end{bmatrix}^T,$$

the output is the quadrotor's positions and yaw angle:

$$\boldsymbol{y} = \begin{bmatrix} x^V & y^V & z^V & \psi \end{bmatrix}^T,$$

the control signal is defined as the squared rotor speeds

$$\boldsymbol{u} = \begin{bmatrix} \omega^2_{m_1} & \omega^2_{m_2} & \omega^2_{m_3} & \omega^2_{m_4} \end{bmatrix}^T,$$

and the state-space realization of the quadrotor plant is

$$\boldsymbol{A}_m = \begin{bmatrix} \boldsymbol{o}_{3\times3} & \boldsymbol{I}_{3\times3} & \boldsymbol{o}_{3\times6} \\ \boldsymbol{o}_{3\times6} & \boldsymbol{A}_{m1} & \boldsymbol{o}_{3\times6} \\ \boldsymbol{o}_{3\times3} & \boldsymbol{o}_{3\times6} & \boldsymbol{I}_{3\times3} \\ \boldsymbol{o}_{3\times3} & \boldsymbol{o}_{3\times3} & \boldsymbol{o}_{3\times3} \end{bmatrix},$$

$$\boldsymbol{B}_m = \begin{bmatrix} \boldsymbol{o}_{5\times4} \\ \begin{bmatrix} b_t/m & b_t/m & b_t/m & b_t/m \end{bmatrix} \\ \boldsymbol{B}_{m1} \\ \boldsymbol{o}_{3\times4} \end{bmatrix},$$

$$\boldsymbol{C}_m = \begin{bmatrix} I_{3\times3} & \boldsymbol{o}_{3\times9} \\ \boldsymbol{o}_{1\times8} & \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \end{bmatrix},$$

where

$$\boldsymbol{A}_{m1} = \begin{bmatrix} 0 & g & 0 \\ -g & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \ \boldsymbol{B}_{m1} = \begin{bmatrix} 0 & -d_{mm}b_t/I_{xx} & 0 & d_{mm}b_t/I_{xx} \\ -d_{mm}b_t/I_{yy} & 0 & d_{mm}b_t/I_{yy} & 0 \\ k_d/I_{zz} & -k_d/I_{zz} & k_d/I_{zz} & -k_d/I_{zz} \end{bmatrix}.$$

It should be noted that the control signals are the squared rotor speeds $\omega^2_{m_{1,2,3,4}}$. As each rotor can only rotate in one direction, only the positive square root of $\omega^2_{m_{1,2,3,4}}$ is used as each rotor's speed reference.

The nonlinear mathematical model of the quadrotor is implemented for simulation in MATLAB and Simulink. The quadrotor model [Corke, 2011] used in this simulation is named as TC2.0. It's parameters are given in Table 6.1, which are different from quadrotor TC1.0 or TC3.0.

The quadrotor TC2.0 centralized MPC parameters are listed in Table 6.2. The weighting matrix $\boldsymbol{Q}$ and $\boldsymbol{R}$ are:

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{o}_{12\times12} & \boldsymbol{I}_{12\times4} \\ \boldsymbol{o}_{4\times12} & 1e5\boldsymbol{I}_{4\times4} \end{bmatrix}$$

$$\boldsymbol{R} = \boldsymbol{I}_{4\times4}$$

Table 6.1: TC2.0 parameters

Parameter	Value	Unit
g	9.81	m/s^2
m	0.4698	kg
d_{mm}	0.3150	m
b_t	$1.31e^{-5}$	N/A
k_d	$1.06e^{-7}$	N/A
I_{xx}	0.082	kgm^2
I_{yy}	0.082	kgm^2
I_{zz}	0.149	kgm^2

.

As the quadrotor has a symmetrical structure, a, N, u^{min} and u^{max} for the four control signals u_1 to u_4 are chosen to be the same. Constraints are applied on the squared rotor speeds $\omega^2_{m_1}$ to $\omega^2_{m_4}$. The upper bound is chosen as the motor's maximum squared speed, and the lower bound is chosen so that enough lift force is provided to keep the quadrotor from descending too fast.

Table 6.2: Quadrotor TC2.0 centralized MPC control parameters

Parameter	Δt	N_p	N_c	a	N	u^{max}	u^{min}
value	$1ms$	500	10	0.5	10	$41 \times 10^4 rad^2/s^2$	$7.34 \times 10^4 rad^2/s^2$

The Simulink model of the centralized quadrotor MPC control is shown in Figure6.2. The detailed settings of the Quadrotor Model block can be found in Appendix C.

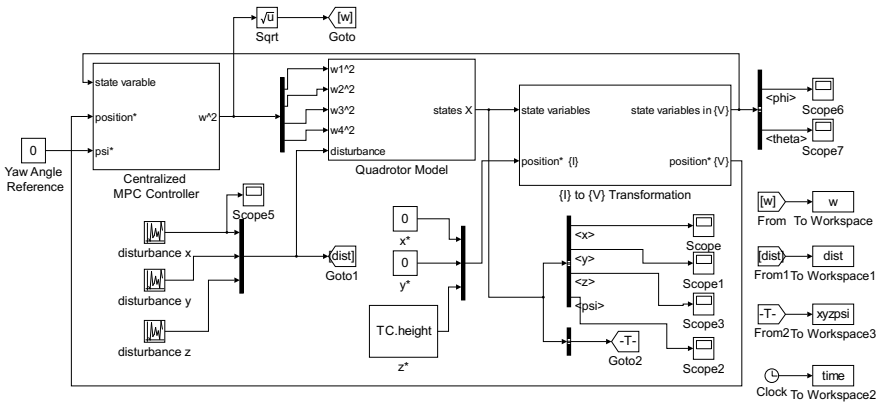
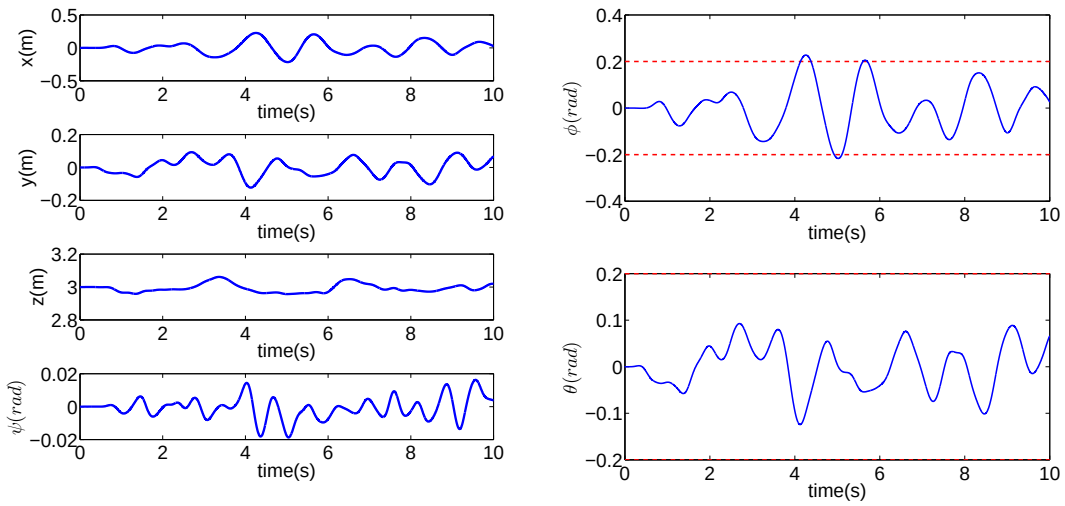


Figure 6.2: Quadrotor TC2.0 centralised MPC control Simulink model

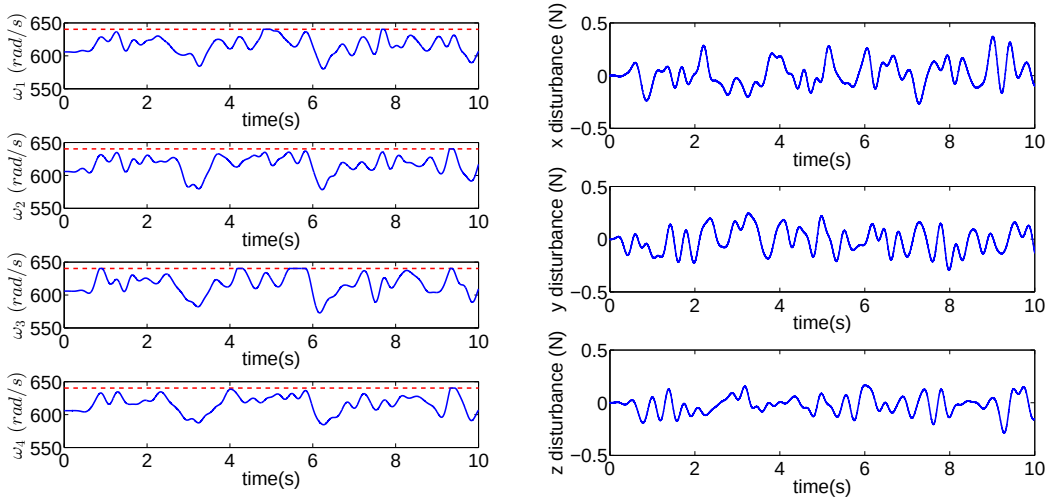
Simulation results are given in Figure 6.3 and Figure 6.4. This simulation shows a scenario that the quadrotor is trying to maintain its position under external disturbance. The quadrotor's hovering position or hovering references are: $x^* = 0$, $y^* = 0$, $z^* = 3$ and $\psi^* = 0$. As it can be seen in Figure 6.4(a), the external disturbances acting on the quadrotor's x and y axis are within ± 0.5 and the disturbance on z axis is within $\pm 0.3N$. In Figure 6.3, it is shown that the quadrotor's positions slightly drift within $\pm 0.2m$ around it x, y and z hovering positions. The quadrotor's yaw angle varies within $\pm 0.02rad$ around its $z-$axis. Figure 6.4(b) shows that the speeds of the quadrotor's four rotors are limited within the constraints.



(a) positions

(b) Euler angles

Figure 6.3: Quadrotor TC2.0 positions and Euler angles under centralised MPC control. Key: dotted line-Euler angle constraints; solid line-Position and Euler outputs



(a) control signals

(b) disturbances

Figure 6.4: Quadrotor TC2.0 rotor speed under centralised MPC control and external disturbance.Key: dotted line-motor speed constraints; solid line-actual motor speed or disturbance

6.4 Quadrotor TC2.0 Cascade MPC Control

A cascade MPC controller is designed to control the quadrotor's positions and yaw angle. As it can be seen in Figure 6.5, the quadrotor plant is divided into the primary and the secondary plant and two MPC controllers are designed accordingly.

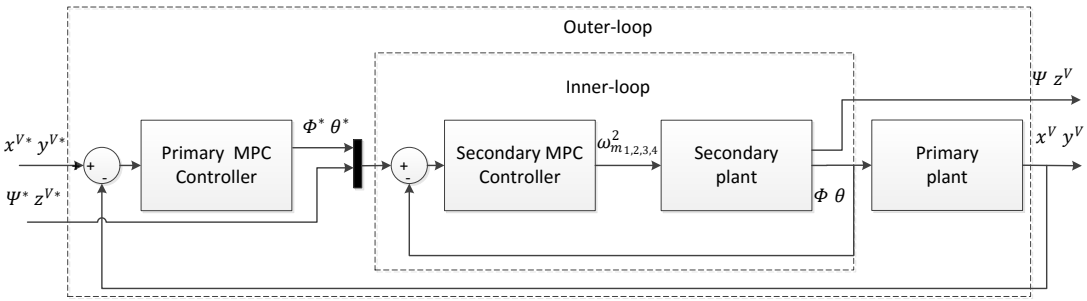


Figure 6.5: Quadrotor TC2.0 Cascade MPC Control diagram

The primary plant's state variable is

$$\boldsymbol{x}_m = \begin{bmatrix} x^V & y^V & \dot{x}^V & \dot{y}^V \end{bmatrix}^T,$$

the output signal is

$$\boldsymbol{y} = \begin{bmatrix} x^V & y^V \end{bmatrix}^T,$$

the control signal is

$$\boldsymbol{u} = \begin{bmatrix} \phi & \theta & T \end{bmatrix}^T,$$

and the primary plant's state-space matrices are

$$\boldsymbol{A}_m = \begin{bmatrix} \boldsymbol{o}_{2\times 2} & I_{2\times 2} \\ \boldsymbol{o}_{2\times 2} & \boldsymbol{o}_{2\times 2} \end{bmatrix}, \ \boldsymbol{B}_m = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & g \\ -g & 0 \end{bmatrix}, \ \boldsymbol{C}_m = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

The primary system's output signals are the quadrotor's x and y positions and the control signals are roll and pitch angles. Constraints are applied on ϕ^* and θ^*, which are the control signals of the primary system and also the reference signals for the secondary system.

The secondary plant's state variable is

$$\boldsymbol{x} = \begin{bmatrix} \phi & \theta & \psi & z & \dot{\phi} & \dot{\theta} & \dot{\psi} & \dot{z}^V \end{bmatrix}^T,$$

the output signal is

$$\boldsymbol{y} = \begin{bmatrix} \phi & \theta & \psi & z^V \end{bmatrix}^T,$$

the control signal is

$$\boldsymbol{u} = \begin{bmatrix} \omega_{m_1}^2 & \omega_{m_2}^2 & \omega_{m_3}^2 & \omega_{m_4}^2 \end{bmatrix}^T,$$

and the secondary plant's state-space matrices are

$$\boldsymbol{A}_m = \begin{bmatrix} \boldsymbol{o}_{4\times4} & \boldsymbol{I}_{4\times4} \\ \boldsymbol{o}_{4\times4} & \boldsymbol{o}_{4\times4} \end{bmatrix}, \ \boldsymbol{B}_m = \begin{bmatrix} \boldsymbol{o}_{4\times4} \\ \boldsymbol{B}_{m1} \end{bmatrix}$$
$$\boldsymbol{C}_m = \begin{bmatrix} \boldsymbol{I}_{4\times4} & \boldsymbol{o}_{4\times4} \end{bmatrix}$$

where

$$\boldsymbol{B}_{m1} = \begin{bmatrix} 0 & -d_{mm}b_t/I_{xx} & 0 & d_{mm}b_t/I_{xx} \\ -d_{mm}b_t/I_{yy} & 0 & d_{mm}b_t/I_{yy} & 0 \\ k_d/I_{zz} & -k_d/I_{zz} & k_d/I_{zz} & -k_d/I_{zz} \\ b_t/m & b_t/m & b_t/m & b_t/m \end{bmatrix}.$$

The secondary plant's output signals ϕ and θ are fed into the primary plant as the primary control signals. The other two output signals ψ and z^V of the the secondary plant are controlled by the secondary controller to follow the reference signals ϕ^* and θ^*. The quadrotor's altitude z^V is included in the secondary plant in order to achieve an unique solution of $\omega^2_{m_{1,2,3,4}}$. Constraints are exerted on the squared rotor speeds.

The Simulink model of the centralized quadrotor MPC control is shown in Figure 6.6. The MPC parameters in this simulation are summarized in Table 6.3 and Talbe

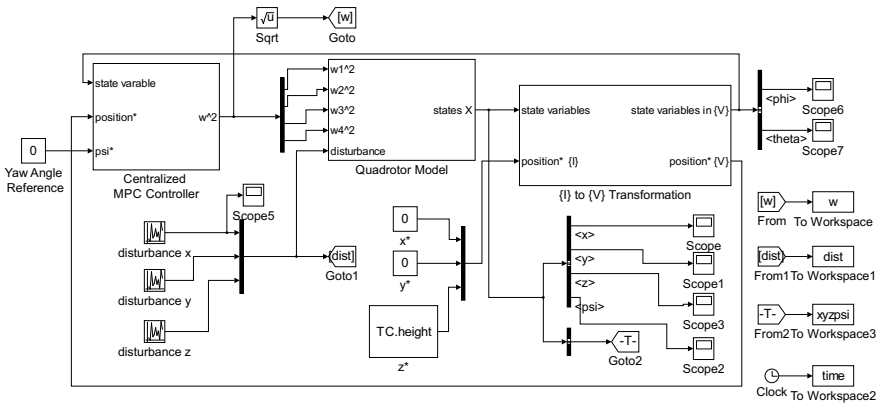


Figure 6.6: Quadrotor TC2.0 centralised MPC control Simulink model

6.4. The position-loop weighting matrix $\boldsymbol{Q}$ and $\boldsymbol{R}$ are:

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{o}_{4\times4} & \boldsymbol{I}_{4\times2} \\ \boldsymbol{o}_{2\times4} & 0.01\boldsymbol{I}_{2\times2} \end{bmatrix}$$
$$\boldsymbol{R} = 1000\boldsymbol{I}_{2\times2}$$

. The secondary-loop weighting matrix $\boldsymbol{Q}$ and $\boldsymbol{R}$ are:

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{o}_{8\times8} & \boldsymbol{I}_{8\times4} \\ \boldsymbol{o}_{4\times8} & 0.01\boldsymbol{I}_{4\times4} \end{bmatrix}$$
$$\boldsymbol{R} = 10\boldsymbol{I}_{4\times4}$$

. The quadrotor's parameters are given in Table 6.1. In this simulation the sampling rate for primary and secondary loops are set to be the same. In hardware implementation the primary loop's sampling interval can be larger than the secondary loop to reduce computational burden. The constraints for the primary control signals are chosen to be $\pm0.2rad$. It should be noted here the constants on the primary control signals can only limit the Euler angle set points, the actually Euler angles can violate these constraints. However, applying constraints on the Euler angle set points makes the quadrotor works under small Euler angles at most of the time. For the secondary loop control signals, the upper bound is chosen as the motor's maximum squared speed, and the lower bound is chosen so that enough lift force is provided to keep the quadrotor from descending too fast.

Table 6.3: Quadrotor TC2.0 cascade MPC primary controller parameters

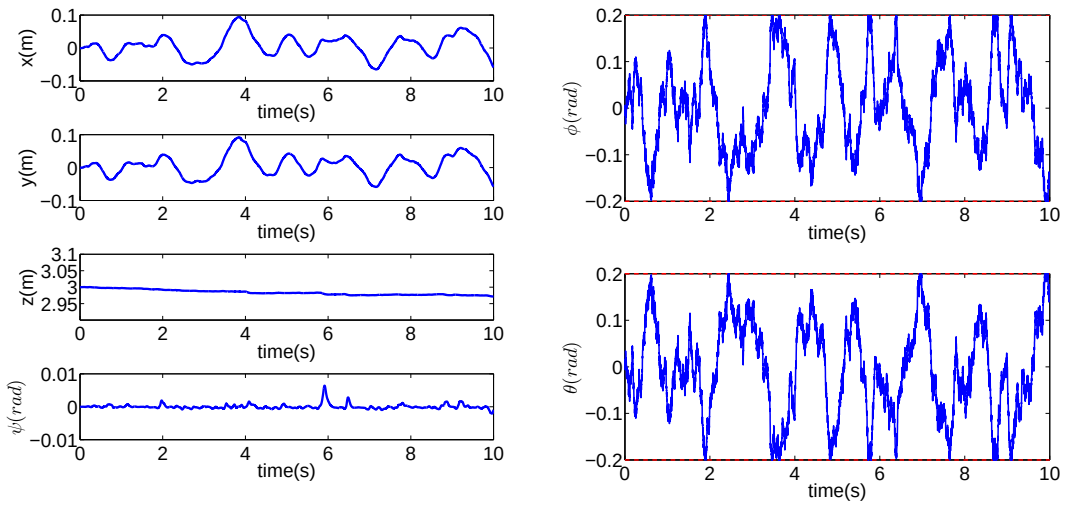
Parameter	Δt	N_p	N_c	a	N	u^{max}	u^{min}
value	$1ms$	500	10	0.5	10	$0.2rad$	$-0.2rad$

Table 6.4: Quadrotor TC2.0 cascade MPC secondary controller parameters

Parameter	Δt	N_p	N_c	a	N	u^{max}	u^{min}
value	$1ms$	500	10	0.5	10	$45\times10^4 rad^2/s^2$	$7.3\times10^4 rad^2/s^2$

The simulation results are shown in Figure 6.7 and 6.8. This simulation shows the same scenario as the centralized MPC control case. The external disturbances are the same as the centralized MPC case. The quadrotor is trying to maintain its position

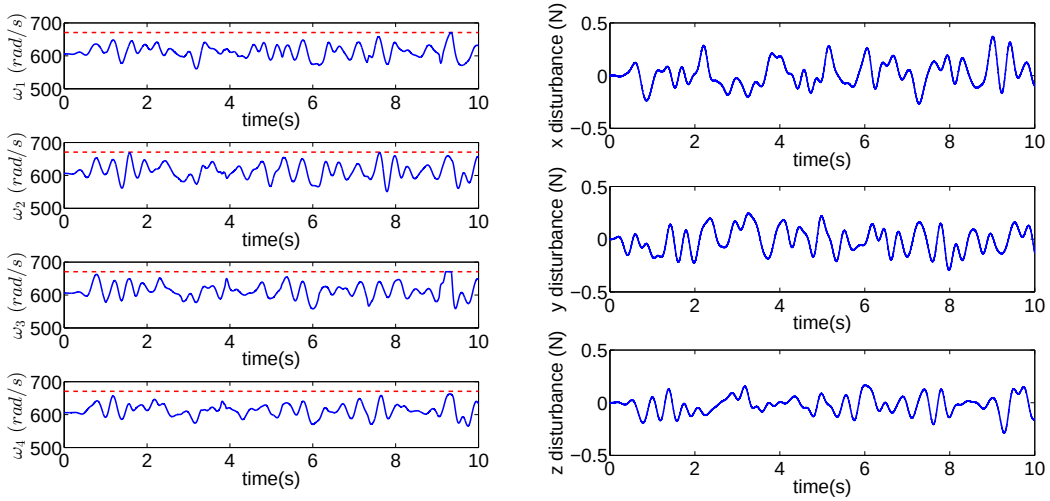
under external disturbance. As shown in Figure 6.8(b), the external disturbance is within $\pm 2.6N$ on the quadrotor's x and y axis and within $\pm 1.4N$ on z axis. In Figure 6.7(a), it is shown that the quadrotor slightly drifts within $\pm 0.1m$ around it x and y hovering positions and within $\pm 0.02rad$ around its $z-$axis. The yaw angle is also kept to be very small between $\pm 0.01rad$. It should be emphasized that the roll and pitch angle, as shown in Figure 6.7(b), are limited within $\pm 0.2rad$.



(a) positions

(b) Euler angles

Figure 6.7: Quadrotor TC2.0 positions and Euler angles under cascade MPC control. Key: dotted line-Euler angle constraints; solid line-Position and Euler outputs



(a) control signals

(b) disturbances

Figure 6.8: Quadrotor rotor TC2.0 speed under cascade MPC control and external disturbance. Key: dotted line-motor speed constraints; solid line- actual motor speed or disturbance

6.5 Comparison of Quadrotor MPC Control Architectures

Several observations can be made from the above simulations on two quadrotor MPC control architectures. Firstly, both MPC control architectures are capable of handling the constrains on the inner-loop control signals, namely the rotor speeds. However, only the cascade MPC is capable of handling the constraints of the outer-loop's intermediate control signals, namely the the roll and pitch angles. Secondly, the cascade MPC shows better disturbance rejection capability than the centralized MPC. This can be seen from the variations on the three axis positions and the yaw angle. In Table 6.5 , we compare the mean squared errors (MSE) of TC2.0's three-axis position outputs. The MSE is calculated using (4.38). Additionally, the cascade PID controller is also tested on TC2.0 to compare with the performance of MPC controllers. As it can be seen, the cascade PID controller offers similar control results as cascade MPC. However, when constraints are present, cascade MPC should be chosen.

Apparently, cascade MPC offers smaller position output MSE than centralized

Table 6.5: TC2.0 Position Outputs MSE

	Centralized MPC	Cascade MPC	Cascade PID
x	9.3e-3	1.4e-3	2.1e-3
y	2.5e-3	6.36e-4	7.4e-4
z	7.2e-4	3.9e-7	3.2e-7

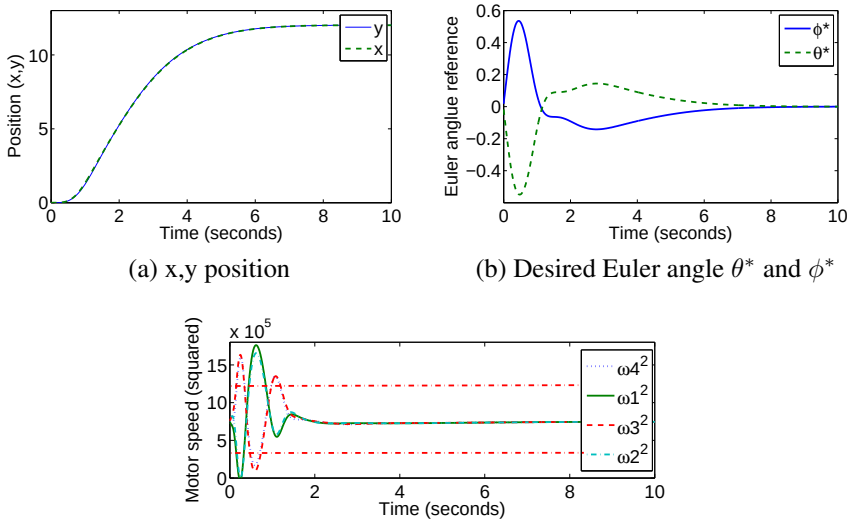
MPC in all three axes. As the two simulations are conducted under the same disturbance levels, it reveals that the cascade MPC possesses better disturbance rejection capability than the centralized MPC. That is because in the cascade MPC control architecture, before all disturbances enter the outer-loop to affect the output signals, they are firstly handled and reduced by the inner-loop controller.

6.6 Constraints Analysis

In this section, we compare different constraint handling configurations in the cascade MPC control architecture. In the cascade MPC, the constraints on the inner-loop control signals are used to limit the actuator outputs, so these constraints must be applied. In additional to that, the cascade MPC control is capable of handling the outer-loop's control signals, which are also the intermediate reference signals for the inner-loop system. As it has been discussed before, applying constraints on those intermediate reference signals can help the system to work in its normal operation conditions and avoid entering the nonlinear region. Other benefits of using the cascade MPC to handle intermediate constraints are discussed in the following simulations. The rated motor speed in this group of simulations is $1100\ rad/s$. In order to prevent the quadrotor from descending dramatically, the minimum motor speed is set to be $600\ rad/s$. The control signals in this group of simulations are the squared motor speeds, so the inner-loop control signal constraints are $3.6e5 \leq \omega_i^2 \leq 12.1e5$. The intermediate inner-loop reference signals (outer-loop control signals) are the Euler angles roll and pitch. The desired Euler angle constraints are $-0.2 \leq \ \phi, \theta \leq 0.2$ to ensure a good region. In addition to QP, imposing pure saturation on the signals is another way to handle constraints. In the following simulations, we also compare the the difference between using QP and pure saturation to handle constraints.

In the first simulation, no constraint is imposed on the angle control loop or position

control loop. As it can be seen in Figure 6.9, position x and y reach the set-pints at $6s$ and desired Euler angles vary between $-0.6\ rad$ to $+0.6\ rad$. The motor speeds are out of the range of the rated motor speed.



(a) x,y position

(b) Desired Euler angle θ^* and ϕ^*

(c) Desired squared motor velocities(horizonal dotted lines the motor speed constraints)

Figure 6.9: Quadrotor TC2.0 position control without constraints.

In the second simulation, pure saturation is used to limit motor speed. As it can be seen on Figure 6.10, the system becomes unstable.

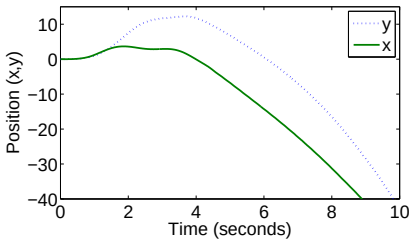


Figure 6.10: Quadrotor TC2.0 position control with pure saturation constraints on the motor speed

In the third simulation, pure saturation is exerted on the desired motor speed and the desired Euler angles. As it can be seen in Figure 6.11, the settling time of of the position control is very long.

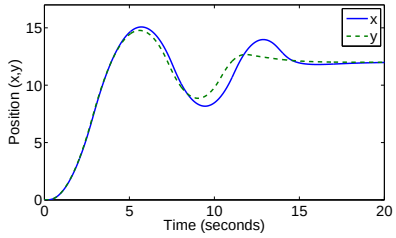
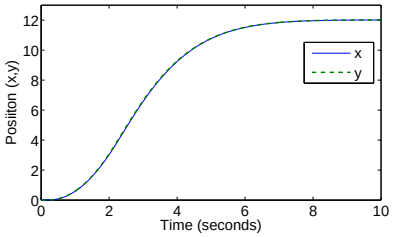
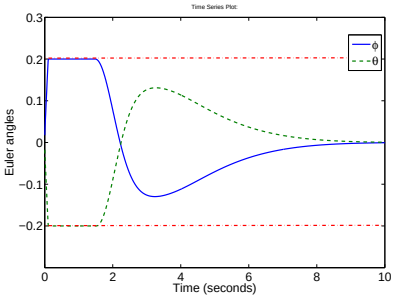


Figure 6.11: Quadrotor TC2.0 position control with pure saturation constraints on the motor speed and desired Euler angles

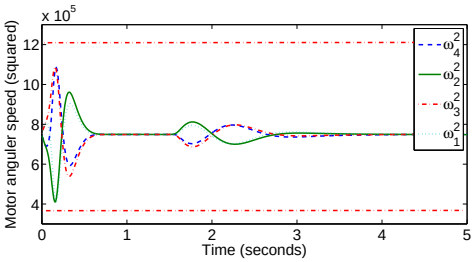
In the fourth simulation, the MPC angle controller uses QP to optimally determine the desired Euler angles under constraints. As it can be seen in Figure 6.12, the quadrotor's response is almost the same the the one without constraints. Also, the desired Euler angles are very small and the desired motor speeds are within the motor speed limits.



(a) x,y position



(b) Desired Euler angle θ^* and ϕ^* (horizonal dotted lines the Euler angle constraints)



(c) Desired squared motor velocities (horizonal dotted lines the motor speed constraints)

Figure 6.12: Quadrotor TC2.0 position control without constraints

6.7 Summary

In this chapter, two MPC control architectures are employed on the quadrotor system. To compare their performances, two groups of simulations are performed. In the first group of simulations, the cascade and centralized MPC controllers are tested in the scenario that the quadrotor maintains its hovering position with little drifting under large disturbances. Even though both MPC controllers can stabilize the system and handle actuator constraints, the cascade MPC shows better disturbance rejection capability. The mean squared errors of the output signals of cascade MPC are much smaller than the centralized MPC controller.

The other group of simulations are conducted using the cascade MPC with different constraint configurations. The key feature of cascade MPC is that constraints can be applied on the intermediate variables, namely ϕ^* and θ^*, to prevent these Euler angles becoming too large. The simulation results show the performance of the cascade MPC controller using QP to handle constraints is better than those one using pure saturation constraints. They reveal that when the quadratic programming is applied to control both the inner-loop and outer-loop systems with constraints, the quadrotor closed-loop system has a fast response and small control signals.

Chapter 7

Nonlinear Predictive Control of Quadrotor and Other Nonlinear Systems

In the previous chapters, the quadrotor controller design is based on the linearized model under the small angle assumption. In Chapter 6, it has shown that the cascade MPC control structure can help to limit the Euler angle set points. However, under large external disturbances, the actual Euler angle can still violate those constraints. As it has been discussed before, if the Euler angles become too large, the controller designed based on the linearized model may not work properly. Even though occasionally violating the small angle assumption generally does not make the quadrotor unstable, it still leads to undesired manurer and unexpected flying trajectory of the quadrotor. To solve above problems, a controller designed based on the quadrotor's nonlinear model is required. In this chapter we will describe the design and application of a novel nonlinear controller called discrete-time one-step-ahead predictive control (DOPC). It will be shown that this nonlinear control technique has simple design procedures and can systematically handle constraints.

The chapter begins with the derivation of the original DOPC control law, followed by the integral DOPC and cascade DOPC design. The robustness analysis is also carried out on the DOPC control systems. Then the quadrotor DOPC controller design is presented. In addition to the quadrotor Euler angle and angular velocity controller

design, the position loop resonant controller design is presented in this chapter.

7.1 Discrete-Time One-step-ahead Predictive Control (DOPC)

This section begins with the derivation of the DOPC control. To eliminate the steady-state errors, integrators are proposed in the original DOPC structure. For system with less inputs than output, the cascade DOPC is required. In the end, the robustness analysis of DOPC is given.

7.1.1 DOPC Design for MIMO Systems

The nonlinear system equation is defined as

$$\dot{\boldsymbol{x}}_m = \boldsymbol{F_M}(\boldsymbol{x_m}) + \boldsymbol{B_m}\boldsymbol{u}, \tag{7.1}$$

where $\boldsymbol{x}_m = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T$ is the state variable, n is the number of state variables, $\boldsymbol{u} = \begin{bmatrix} u_1 & u_2 & \cdots & u_m \end{bmatrix}^T$ is the control signal, m is the number of control signals, $\boldsymbol{F_M}(\boldsymbol{x_m})$ is the nonlinear term. Here, we assume that the number of inputs is larger or equal to the number of states. The discrete form of this nonlinear system is

$$\frac{\boldsymbol{x}(t_{k+1}) - \boldsymbol{x}(t_k)}{\Delta t} \approx \boldsymbol{F_M}(\boldsymbol{x}(t_k)) + \boldsymbol{B}\boldsymbol{u}(t_k), \tag{7.2}$$

where Δt is the sampling interval, t_k is the kth sampling instance. For notational simplicity, in the following equations $\boldsymbol{x}(t_k)$ and $\boldsymbol{u}(t_k)$ will be written as $\boldsymbol{x}(k)$ and $\boldsymbol{u}(k)$. $\boldsymbol{F_M}(\boldsymbol{x}(t_k))$ will be written as $\boldsymbol{F_M}(k)$, but it is still a function of $\boldsymbol{x}(t_k)$.

The objective function J is defined as

$$J = (\boldsymbol{x}^*(k) - \boldsymbol{x}(k+1))^T \boldsymbol{Q} (\boldsymbol{x}^*(k) - \boldsymbol{x}(k+1)) + \boldsymbol{u}(k)^T \boldsymbol{R}\boldsymbol{u}(k), \tag{7.3}$$

where $\boldsymbol{x}^*(k)$ is the reference signal for $\boldsymbol{x}(k)$ at the ith sampling instance and $\boldsymbol{Q}$ and $\boldsymbol{R}$ are weighting matrices for the state variable and control signal respectively. Substitut-

ing equation (7.2) into the objective function yields

$$J = [\boldsymbol{x}^*(k) - \boldsymbol{x}(k) - \Delta t(\boldsymbol{F_M}(k) + \boldsymbol{B}\boldsymbol{u}(k))]^T * \\ \boldsymbol{Q}[(\boldsymbol{x}^*(k) - \boldsymbol{x}(k) - \Delta t(\boldsymbol{F_M}(k) + \boldsymbol{B}\boldsymbol{u}(k))] + \boldsymbol{u}(k)^T \boldsymbol{R}\boldsymbol{u}(k).$$

Letting $\boldsymbol{H}(k) = \boldsymbol{x}^*(k) - \boldsymbol{x}(k) - \Delta t \boldsymbol{F_M}(k)$, the objective function J becomes

$$J = [\boldsymbol{H}(k) - \Delta t \boldsymbol{B}\boldsymbol{u}(k)]^T \boldsymbol{Q}[\boldsymbol{H}(k) - \Delta t \boldsymbol{B}\boldsymbol{u}(k)] + \boldsymbol{u}(k)^T \boldsymbol{R}\boldsymbol{u}(k),$$

which can be rearranged to be the quadratic objective function form:

$$J = \boldsymbol{H}(k)^T \boldsymbol{Q}\boldsymbol{H}(k) - 2\boldsymbol{u}(k)^T \Delta t \boldsymbol{B}^T \boldsymbol{Q}\boldsymbol{H}(k) + \Delta t^2 \boldsymbol{u}(k)^T \boldsymbol{B}^T \boldsymbol{Q}\boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{u}(k)^T \boldsymbol{R}\boldsymbol{u}(k). \tag{7.4}$$

Now this becomes a least squares minimization problem. From the first derivative of the objective function J:

$$\frac{\partial J}{\partial \boldsymbol{u}(k)} = -2\Delta t \boldsymbol{B}^T \boldsymbol{Q}\boldsymbol{H}(k) + 2\Delta t^2 \boldsymbol{B}^T \boldsymbol{Q}\boldsymbol{B}\boldsymbol{u}(k) + 2\boldsymbol{R}\boldsymbol{u}(k),$$

the necessary condition of the minimum J is obtained as

$$\frac{\partial J}{\partial \boldsymbol{u}(k)} = 0.$$

With the assumption that the Hessian matrix $\Delta t^2 \boldsymbol{B}^T \boldsymbol{Q}\boldsymbol{B} + \boldsymbol{R}$ is positive definite, the optimal control signal that will minimize the objective function (7.4) is given by the following solution:

$$\boldsymbol{u}(k) = (\Delta t^2 \boldsymbol{B}^T \boldsymbol{Q}\boldsymbol{B} + \boldsymbol{R})^{-1} * \Delta t \boldsymbol{B}^T \boldsymbol{Q}\boldsymbol{H}(k). \tag{7.5}$$

Substituting $\boldsymbol{H}(k)$ to equation (7.5) gives the expression:

$$\boldsymbol{u}(k) = (\Delta t^2 \boldsymbol{B}^T \boldsymbol{Q}\boldsymbol{B} + \boldsymbol{R})^{-1} * \Delta t \boldsymbol{B}^T \boldsymbol{Q}(\boldsymbol{x}^*(k) - \boldsymbol{x}(k) - \Delta t \boldsymbol{F_M}(k)). \tag{7.6}$$

When the weighting matrix $\boldsymbol{R} = 0$, assuming that $\boldsymbol{B}^T \boldsymbol{Q}\boldsymbol{B}$ is positive definite,

substituting the above control signal to equation (7.1) gives:

$$\boldsymbol{x}(k+1) = \boldsymbol{x}(k)^*.$$

In terms of z-transfer function, for each state variable, a pure time-delay transfer function is used to describe the relationship between the reference and the state variable as:

$$\frac{X_i^*(z)}{X_i(z)} = z^{-1}, \tag{7.7}$$

where $X_i(z)$ is the z transform of one of the state variables. The above equation reveals that the the closed-loop dynamics of each state variable becomes a unit delay. When the weighting matrix is not zero, substituting the control signal to equation (7.1) gives:

$$\begin{aligned}\boldsymbol{x}(k+1) = \boldsymbol{x}(k) + \Delta t \boldsymbol{F_M}(k) + \\ \Delta t^2 \boldsymbol{B}(\Delta t^2 \boldsymbol{B}^T \boldsymbol{Q} \boldsymbol{B} + \boldsymbol{R})^{-1} \boldsymbol{B}^T \boldsymbol{Q}[\boldsymbol{x}(k)^* - \boldsymbol{x}(k) - \Delta t \boldsymbol{F_M}(k)]\end{aligned} \tag{7.8}$$

Assuming $\boldsymbol{\Lambda} = \Delta t^2 \boldsymbol{B}(\Delta t^2 \boldsymbol{B}^T \boldsymbol{Q} \boldsymbol{B} + \boldsymbol{R})^{-1} \boldsymbol{B}^T \boldsymbol{Q}$, the above equation becomes:

$$\boldsymbol{x}(k+1) = (\boldsymbol{I} - \boldsymbol{\Lambda})\boldsymbol{x}(k) + \boldsymbol{\Lambda}\boldsymbol{x}(k)^* + (\boldsymbol{I} - \boldsymbol{\Lambda})(\Delta t \boldsymbol{F_M}(k)). \tag{7.9}$$

When Δt is small enough, the quantity $\Delta t(\boldsymbol{I} - \boldsymbol{\Lambda})\boldsymbol{F_M}(k)$ is sufficiently small and the closed-loop system becomes

$$\boldsymbol{x}(k+1) \approx (\boldsymbol{I} - \boldsymbol{\Lambda})\boldsymbol{x}(k) + \boldsymbol{\Lambda}\boldsymbol{x}(k)^*. \tag{7.10}$$

The dynamic characteristics of this closed-loop system are all determined by the eigenvalues in matrix $\boldsymbol{I} - \boldsymbol{\Lambda}$. As $\boldsymbol{\Lambda}$ is a function of Δt and the weighting matrix $\boldsymbol{Q}$, $\boldsymbol{R}$, we can change the response speed of the closed-loop system by modifying $\boldsymbol{Q}$ and $\boldsymbol{R}$ and Δt. Generally, increasing $\boldsymbol{Q}$ or decreasing $\boldsymbol{R}$ will make closed-loop eigenvalues move towards the origin and thus offer the system a faster response but it also requires large control signals.

Regarding to the choice of the sampling rate, the best engineering choice is the slowest-possible sample rate that still meets all performance specifications such as closed-loop bandwidth, disturbances rejection [Franklin et al., 1994]. Typically, the

sampling rate $\frac{1}{\Delta t}$ is chosen to be 5 to 10 times the closed-loop bandwidth. In reality, this range of sampling rates will result in more or less performance degradation from the desired closed-loop performance we aimed at. To make the sampling effect truly negligible, the sampling rate should be 20 times the closed-loop bandwidth. For the DOPC controller, once the sampling rate is chosen, we should tune $\boldsymbol{Q}$ and $\boldsymbol{R}$ to make the closed-loop system meet the design specifications.

As DOPC is an optimization based control law, the quadratic programming introduced in chapter6 can be employed to deal with constraints in DOPC control.

Comparing the corresponding coefficients in (6.26) and (7.4), we get

$$\boldsymbol{E} = \Delta t^2 \boldsymbol{B}^T \boldsymbol{Q} \boldsymbol{B} + \boldsymbol{R}, \tag{7.11}$$

$$\boldsymbol{F} = -\Delta t \boldsymbol{B}^T \boldsymbol{Q} \boldsymbol{H}(i). \tag{7.12}$$

Similarly, Hildreth's Quadratic Programming can be utilized to solve the quadratic programming problem.

7.1.2 Integral DOPC

In order to eliminate the steady-state errors, a discrete-time integral controller will be added on the original DOPC controller. There are two mainstream approaches to design a control system with integral action. The first approach is to embed the integrator into the controller and the second approach is to estimate a constant input disturbance using an observer followed by subtraction of this constant disturbance from the control signal. The details of the latter approach can be found in [Goodwin et al., 2001]. The first approach is the most widely used method in the applications. Among them are PID controllers that have embedded the integral function in the controller and the model predictive controllers with incremental models [Wang, 2009]. The integral controller is given as $\frac{K_i}{1-z^{-1}}$. The DOPC controller with integral action is illustrated in Figure 7.1. This controller is named as I-DOPC. The remaining problem is how to design the integral gain K_i. The whole system in Figure 7.1 can be considered as a cascade system with two control loops. The inner closed-loop system consists of the nonlinear plant and the DOPC controller. The outer closed-loop system is the inner-loop plus the integral controller. The inner-loop's reference signal is the outer-loop's

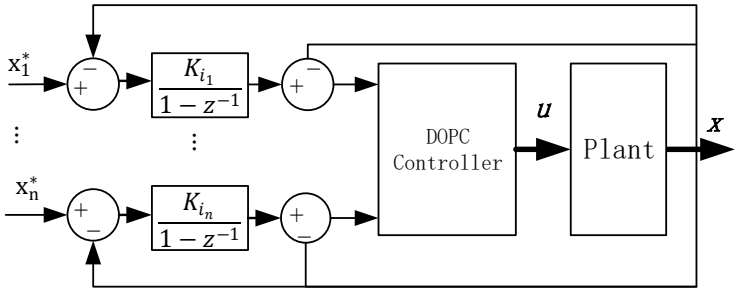


Figure 7.1: DOPC control system with integrator

desired control signal. The inner-loop's output signal is the outer-loop's actual control signal. For a cascade DOPC system, it has been shown before that if the weighting factor $R = 0$, its inner closed-loop system becomes one sample delay. It is apparent that the open-loop transfer function for the outer-loop system includes the integral controller $\frac{K_i}{1-z^{-1}}$ together with the time delay z^{-1} from the inner closed-loop system. Hence, the outer closed-loop system has the transfer function:

$$\frac{X(z)}{X^*(z)} = \frac{K_i z^{-1}}{1 - z^{-1} + K_i z^{-1}}, \tag{7.13}$$

, which can be rearranged to be

$$\frac{X(z)}{X^*(z)} = \frac{K_i}{z - (1 - K_i)}, \tag{7.14}$$

To determine K_i, firstly we choose the desired continuous-time closed-loop time constant ϵ_{ss} and hence the continuous-time closed-loop pole will be $s = -\frac{1}{\epsilon_{ss}}$. Secondly, we calculate the discrete-time closed-loop pole as $z = e^{-\Delta t a_{cl}}$. Thirdly, we calculate K_i as

$$K_i = 1 - e^{-\frac{\Delta t}{\epsilon_{ss}}} \tag{7.15}$$

Another benefit of I-DOPC is that the closed-loop dynamics of each channel can be tuned independently by choosing different K_i, as the inner-loop system is already decouple. It should be noted that when the weighting matrix $\boldsymbol{R}$ is not zero, the inner closed-loop dynamics is not a unit delay so the pole location is not $1 - K_i$ any more. However, using a non-zero weighing matrix $\boldsymbol{R}$ makes the I-DOPC design more com-

plex without bringing in much benefits. As a result, it is proposed that in I-DOPC controller design the inner-loop's weighing matrix $\boldsymbol{R}$ should be chosen to be zero and $\boldsymbol{Q}$ to should be an identity matrix.

7.1.3 Cascade DOPC

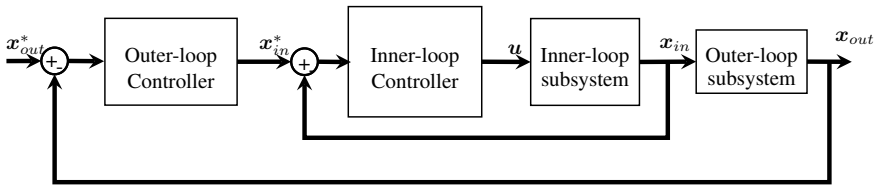


Figure 7.2: Cascade DOPC structure

The design of DOPC requires the number of control signals to be equal to or greater than the state variables. However, it is very common to encounter systems with less control signals than their state variables. For those systems, some state variables are effectively controlled by other state variables rather than the external control signals. For example, a DC motor's speed is controlled by its armature voltage, which is an external control signal, whereas the motor's position is controlled by the motor's speed, which is a state variable. For those systems with less control signals than state variables, a cascaded control structure can be used in DOPC controller design. The basic idea of cascade control is to break a system into some subsystems and design controllers for each subsystem separately. For each subsystem, their control signals need to be equal to or greater than their state variables.

Normally, in a cascade control system, the inner closed-loop dynamics needs to be much faster than the outer-loop dynamics so that the inner closed-loop dynamics can be ignored when designing the outer-loop controller. For the cascade DOPC control system, as the closed-loop dynamics is mainly determined by the sampling interval Δt, the inner-loop DOPC requires a shorter sampling interval to get a fast response and outer-loop DOPC can use a longer sampling interval. Normally, the inner-loop sampling interval should be ten times shorter than the outer-loop. By using dual sampling rates, the computational cost for the entire control system is reduced.

Assuming the nonlinear system is given as (7.1) and it can be divided into the following two subsystems:

$$\dot{\boldsymbol{x}}_{out} = \boldsymbol{F}_{out}(\boldsymbol{x}) + \boldsymbol{B}_{out}\boldsymbol{x}_{in} \tag{7.16}$$

$$\dot{\boldsymbol{x}}_{in} = \boldsymbol{F}_{in}(x) + \boldsymbol{B}_{in}\boldsymbol{u} \tag{7.17}$$

(7.16) is called outer-loop system and (7.17) is called inner-loop system. The inner-loop control signal is the external control signal u in the original system of equation (7.1). The inner-loop system's output signals become the outer-loop system's control signals. The block diagram of cascade DOPC control is given in Figure 7.2. The DOPC controller can be used as the inner-loop controller to deal with the nonlinearity and offer the inner closed-loop a fast response. Integrator is not necessary for inner-loop system as steady-state errors will not affect the outer-loop system much. Another benefit of using DOPC as the inner-loop controller is that constraints can be handled optimally by using quadratic programming. As the inner-loop control signals are normally the actual physical signals such as forces or torques generated by the actuator, adding constraints is necessary. For the outer-loop controller, if the outer-loop system contains strong nonlinearity, I-DOPC controller can be used to deal with the nonlinearity and eliminate steady-state errors as well. However, it is very common that a physical system's outer-loop dynamics is linear, where PI controllers can be used. For the outer-loop system, constraints are not necessary because the outer-loop control signals are the reference signals of the inner-loop system instead of the actual control signals generated by actuators.

To design a cascade DOPC control system, the inner-loop system should be tuned first. It should be tuned as fast as possible subject to the actuators' limitation. In other words, the DOPC controller gain should be tuned as high as possible until the control signals start to exceed the actuators' limitation. The outer-loop closed-loop system should be tuned to be slower than the inner-loop closed-loop system so that the inner-loop dynamics can be ignored. Then the outer-loop controller is tuned by any traditionally control system design methods to meet system specifications, such as overshot, rise time and settling time.

7.1.4 DOPC Robustness Analysis

Let us consider a nonlinear system described by the following equation:

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \end{bmatrix} = \begin{bmatrix} a_1 x_2 x_3 \\ a_2 x_1 x_3 \\ a_3 x_1 x_2 \end{bmatrix} + \begin{bmatrix} b_1 & 0 & 0 \\ 0 & b_2 & 0 \\ 0 & 0 & b_3 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$

where x_1, x_2, x_3 are state variables, u_1, u_2, u_3 are control signals. $a_1 = 27.5,\ a_2 = 11.2500,\ a_3 = -34.4828, b_1 = 6400,\ b_2 = 3200,\ b_3 = 4414.$

In this example we assume that there are unmodelled dynamics on control signals. The unmodelled dynamics on each control signal is described as

$$G(s) = \frac{u_i}{u_i^*} = \frac{K_{ss}}{\epsilon_m s + 1}, \tag{7.18}$$

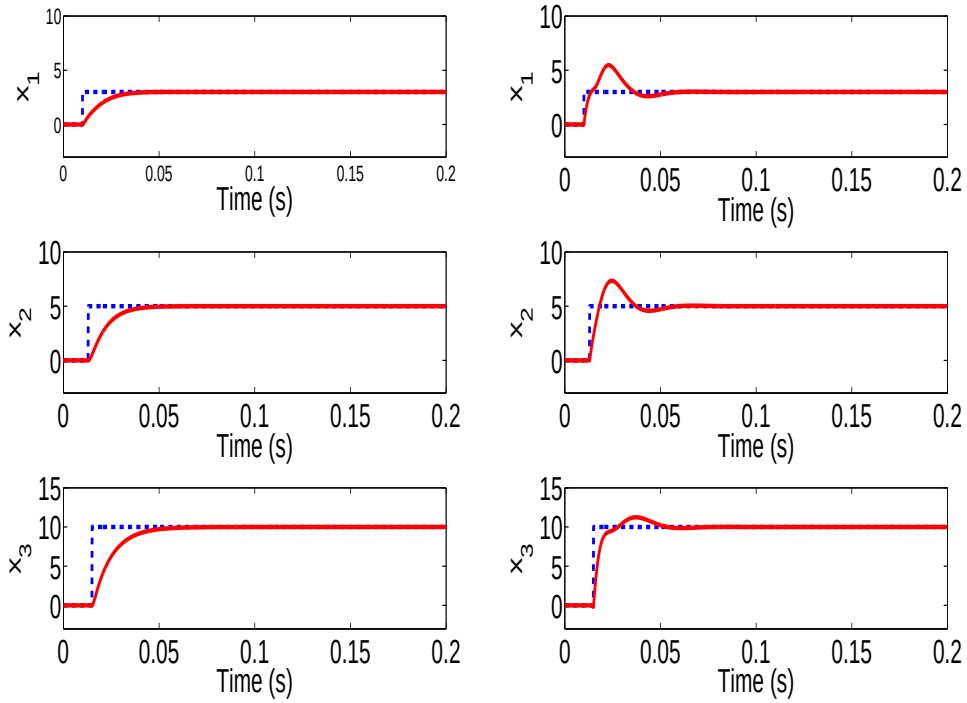
where K_{ss} is the steady-state gain, ϵ_m is the time constant, u_i and u_i^* are the actual and desired control signal for each channel.

By adding control signals' dynamics to the original system, the dynamic model becomes

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{u_1} \\ \dot{u_2} \\ \dot{u_3} \end{bmatrix} = \begin{bmatrix} a_1 x_2 x_3 \\ a_2 x_1 x_3 \\ a_3 x_1 x_2 \\ -u_1/\epsilon_m \\ -u_2/\epsilon_m \\ -u_3/\epsilon_m \end{bmatrix} + \begin{bmatrix} b1 & 0 & 0 \\ 0 & b2 & 0 \\ 0 & 0 & b3 \\ \frac{Kss}{\epsilon_1} & 0 & 0 \\ 0 & \frac{Kss}{\epsilon_2} & 0 \\ 0 & 0 & \frac{Kss}{\epsilon_3} \end{bmatrix} \begin{bmatrix} u1 \\ u2 \\ u3 \end{bmatrix}.$$

In this simulation, the weighting matrix $\boldsymbol{R} = I_{3\times3}$ and $\boldsymbol{Q} = I_{3\times3}$. The integral gains for all three channels are the same $K_i = 0.01$. Sampling interval is $\Delta t = 0.0001$. We assume the unmodelled actuator dynamics in three axes are identical and the steady-state gain K_{ss} varies between 0.5 to 5 and the time constant is $\epsilon_m = 0.0001$. Comparisons are made between the DOPC controller and the PI controller.

Figure 7.3 shows step responses of the DOPC system and the PI system without any unmodelled dynamics. Figure 7.4 and Figure 7.5 show step responses of of DOPC system and PI system with different levels of unmodelled dynamics. As we can see



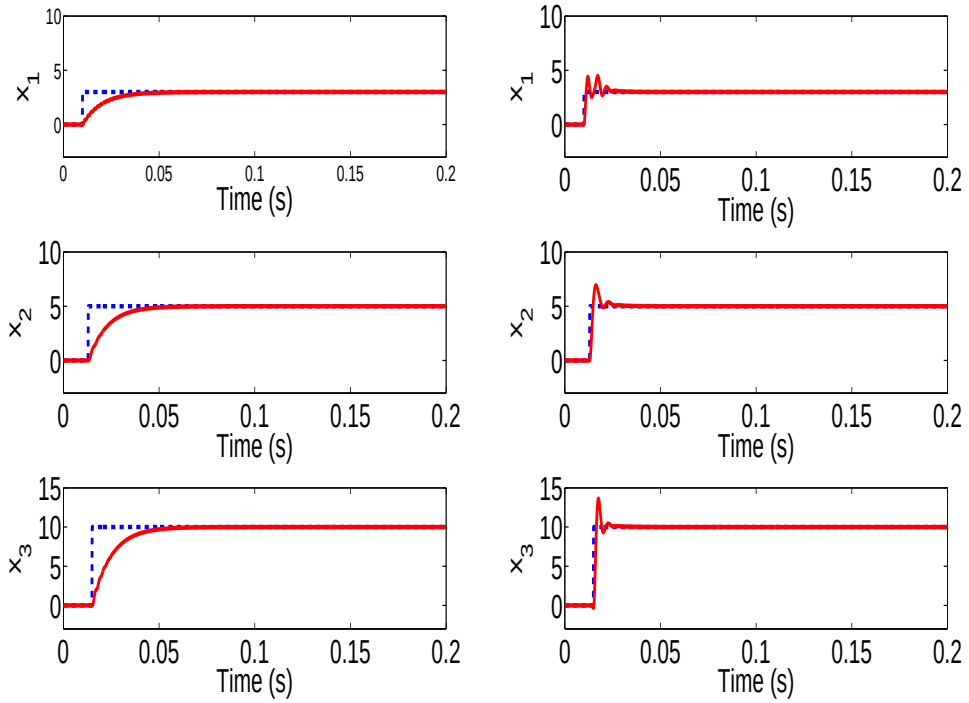
(a) DOPC controller

(b) PI controller

Figure 7.3: I-DOPC and PI robustness comparison without unmodelled actuator dynamics. Key: dotted lines reference signals; solid lines output signals.

.

from the simulations, when K_{ss} changes from 0.5 to 5, there is no obvious change in the output signals of DOPC system. In comparison, the PI system is more sensitive to the unmodelled dynamics. When the unmodelled dynamics's K_{ss} is increased to 5, the PI system's response becomes faster with larger overshot and more oscillation. When K_{ss} is reduced to 0.5, the PI system's performance degrades seriously.

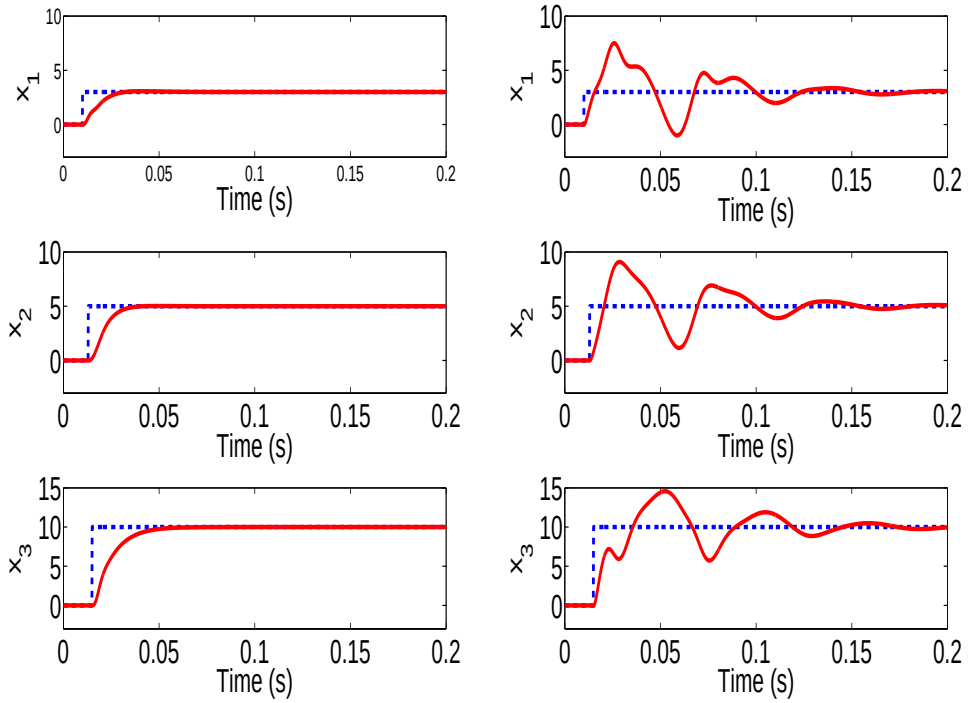


(a) DOPC controller

(b) PI controller

Figure 7.4: I-DOPC and PI robustness comparison with unmodelled actuator dynamics $\epsilon_m = 0.001$ and $K_{ss} = 5$. Key: dotted lines reference signals; solid lines output signals.

.



(a) DOPC controller

(b) PI controller

Figure 7.5: I-DOPC and PI robustness comparison with unmodelled actuator dynamics $\epsilon_m = 0.001$ and $K_{ss} = 0.5$. Key: dotted lines reference signals; solid lines output signals.

.

7.2 Quadrotor Nonlinear Flight Controller Design

To design the quadrotor DOPC controller, the quadrotor dynamics is divided into angle and position subsystems, and the angle subsystem is further divided into angular velocity subsystem and Euler angular subsystem. A cascade control structure is adopted to stabilize those three subsystems, which is illustrated by Figure 7.6. The angular velocity subsystem is controlled by a DOPC controller and is referred as the inner-loop. The inner-loop controller's inputs are the angular velocity feedback and the desired angular velocities. The inner-loop controller's outputs are the desired torques. The angle subsystem is also controlled by a DOPC controller and referred as the middle-loop. The middle-loop controller's inputs are the angle feedback and the desired angles. The middle-loop controller's outputs are the desired angular velocities. The position subsystem is controlled by a PI controller and referred to as the outer-loop. The outer-loop controller's inputs are the position reference and the actual position feedback. The outer-loop controller's outputs are the desired angles. The design of above controllers will be given in the following sections.

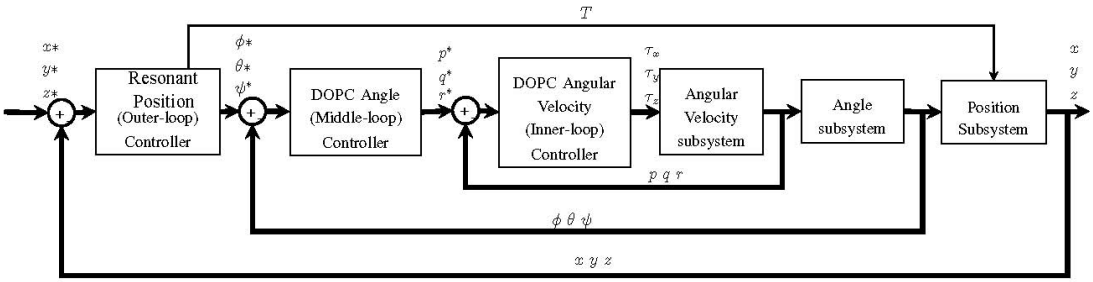


Figure 7.6: Closed-loop control diagram of quadrotor UAV

7.2.1 Angular Velocity Subsystem (Inner-loop) Controller Design

The dynamic equation of angular velocity subsystem is given as

$$\overbrace{\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix}}^{\dot{\boldsymbol{x}}_m} = \overbrace{\begin{bmatrix}(I_{yy}-I_{zz})qr/I_{xx}\\ (I_{zz}-I_{xx})pr/I_{yy}\\ (I_{xx}-I_{yy})pq/I_{zz}\end{bmatrix}}^{\boldsymbol{F}(\boldsymbol{x}_m)} + \overbrace{\begin{bmatrix}1/I_{xx} & 0 & 0\\ 0 & 1/I_{yy} & 0\\ 0 & 0 & 1/I_{zz}\end{bmatrix}}^{\boldsymbol{B}_m}\overbrace{\begin{bmatrix}\tau_x\\ \tau_y\\ \tau_z\end{bmatrix}}^{\boldsymbol{u}}. \tag{7.19}$$

This is a nonlinear system with $[p\ q\ r]^T$ as its state variable and $[\tau_x\ \tau_y\ \tau_z]^T$ as its control signal. As the inner-loop (angular velocity)'s control signals are the torques created by actuators, they need to be limited and hence constraints will be imposed on the inner-loop controller. The optimal control signals with constraints need to be solved iteratively on-line. The reference signals of the angular velocity subsystem are the desired angular velocities p^*, q^* and r^*, which are the control signals of the angle subsystem.

7.2.2 Euler angle subsystem (Middle-loop) controller design

The dynamic equation of Euler angle subsystem is given as

$$\overbrace{\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix}}^{\dot{\boldsymbol{x}}_m} = \overbrace{\begin{bmatrix}1 & S_\phi T_\theta & C_\phi T_\theta\\ 0 & C_\phi & -S_\phi\\ 0 & S_\phi/C_\theta & C_\phi/C_\theta\end{bmatrix}}^{\boldsymbol{B}_m}\overbrace{\begin{bmatrix}p\\ q\\ r\end{bmatrix}}^{\boldsymbol{u}}. \tag{7.20}$$

This is a nonlinear system with $[\phi\ \theta\ \psi]^T$ as the state variable and $[p\ q\ r]^T$ as the control signal. As the middle-loop control signal will become the reference signal of the inner-loop system instead of the actual control signal created by actuators, it is not necessary to limit the middle-loop's control signal. It is emphasized here that the matrix B is not a constant matrix and needs to be updated at the beginning of each sampling instance.

For a cascade DOPC control system, the inner closed-loop dynamics need to be much faster then the outer closed-loop dynamics so that the inner-loop dynamics can be ignored. For the quadrotor cascade controller design, the sampling interval Δt and the weighting matrix Q and R can be tuned to make the angular velocity closed-loop

system (Inner-loop) faster then the Euler angle closed-loop system (Middle-loop). The position closed-loop system (Outer-loop) should have the slowest response speed.

7.2.3 Position subsystem resonant controller design

In the frame $\{V\}$, ψ is equal to zero as the frame is rotating with the quadrotor. The translational dynamics of the quadrotor in $\{V\}$ is give as

$$\begin{bmatrix} \ddot{x}^V \\ \ddot{y}^V \\ \ddot{z}^V \end{bmatrix} = \frac{1}{m} \begin{bmatrix} -TS_\theta C_\phi \\ TS_\phi \\ -TC_\theta C_\phi + mg \end{bmatrix}. \tag{7.21}$$

Assuming

$$u = \begin{bmatrix} u_a \\ u_b \\ u_c \end{bmatrix} = \begin{bmatrix} -TS_\theta C_\phi \\ TS_\phi \\ -TC_\theta C_\phi + mg \end{bmatrix}, \ \bar{u} = \begin{bmatrix} \phi \\ \theta \\ T \end{bmatrix}, \tag{7.22}$$

then equation (7.21) becomes

$$\begin{bmatrix} \ddot{x}^V \\ \ddot{y}^V \\ \ddot{z}^V \end{bmatrix} = \frac{1}{m} \begin{bmatrix} u_a \\ u_b \\ u_c \end{bmatrix} \tag{7.23}$$

It is obvious that (7.23) is a linear system and thus the position subsystem controller may be designed by using any traditional linear control techniques.

It is well known from the internal model control principle that in order to reject a periodic disturbance, or follow a periodic reference signal with zero steady state error, the generator for the disturbance or the reference must be included in the stable closed-loop control system [Francis and Wonham, 1976]. With the embedded mode, the closed-loop feedback control system is designed to be stable, and at the steady-state, the output of the control system will completely track the sinusoidal signal and/or reject a sinusoidal disturbance signal without any steady-state errors. In the literature, this type of controller is also called resonant controller[Liuping Wang, 2015]. However, when applying the resonant position controller on the quadrotor, the position loop controller gain becomes very high, which leads to dramatic changing in Euler angles

and the angle loop controllers need to constantly work in the nonlinear regions. In this case, nonlinear angle loop controllers, such as DOPC control, is required. In the remaining part of this section, the resonant position loop controller is developed and applied on TC2.0, along with the angle loop DOPC controllers.

One of quadrotor applications that requires resonant controllers is quadrotor tracking a circular path wit a constant speed. In that case, the $x-$ and $y-$axis position references are both sinusoidal signals with the same frequency ω_r. The resonant controller is in this form:

$$\frac{k_1 + k_2 z^{-1} + k_3 z^{-2} + k_4 z^{-3}}{1 - 2z^{-1} cos(\omega_r \Delta t) + z^{-2}}. \tag{7.24}$$

All three channels in the position subsystem have the same dynamics as shown in (7.25).

$$\begin{bmatrix} \ddot{x}^V \\ \ddot{y}^V \\ \ddot{z}^V \end{bmatrix} = \frac{1}{m} \begin{bmatrix} -TS_\theta C_\phi \\ TS_\phi \\ -TC_\theta C_\phi + mg \end{bmatrix}. \tag{7.25}$$

The discrete transfer function for each channel is:

$$\frac{\frac{1}{m}\Delta t^2 z^{-1}}{1 - 2z^{-1} + z^{-2}}. \tag{7.26}$$

Hence the closed-loop characteristic equation is

$$(k_1+k_2 z^{-1}+k_3 z^{-2}+k_4 z^{-3})(\frac{1}{m}\Delta t^2 z^{-1})+(1-2z^{-1}cos(\omega_r \Delta t)+z^{-2})(1-2z^{-1}+z^{-2}) = 0, \tag{7.27}$$

where $k_1,\ k_2,\ k_3,\ k_4$ are unknown controller parameters. Assuming the desired closed-loop characteristic equation is in this form:

$$(1 - \delta z^{-1})^4 = 0, \tag{7.28}$$

where δ determines the desired pole location, which should be smaller than 1. Comparing the corresponding coefficients in equation (7.27) and equation (7.28), we can

determine $k_1,\ k_2,\ k_3$, and k_4:

$$k_1 = \frac{-4\delta + 2cos(w_f * \Delta t) + 2}{\frac{1}{m}T^2}$$
$$k_2 = \frac{6\delta^2 - 4cos(w_f * \Delta t) - 2}{\frac{1}{m}T^2}$$
$$k_3 = \frac{-4\delta^3 + 2cos(w_f * \Delta t) + 2}{\frac{1}{m}T^2}$$
$$k_4 = \frac{\delta^4 - 1}{\frac{1}{m}T^2}.$$

Once the control signal $[u_a\ u_b\ u_c]^T$ is achieved, it can be converted to $[\phi\ \theta\ T]^T$ by using Algorithm 1.

Algorithm 1 Algorithm of u to $\bar{u}$ conversion

if $u_c > \frac{1}{2}mg$ **then**
  $u_c = \frac{1}{2}mg$
end if
if $(u_a == 0)\&\&(u_b == 0)$ **then**
  $\theta = 0$
  $\phi = 0$
  $T = mg - u_c$
else
  if $(u_a) == 0)\&\&(u_b \neq 0)$ **then**
    $\theta = 0$
    $\phi = arctan(-u_b/(u_c - mg))$
    $T = u_b/sin\phi$
  end if
else
  if $(u_a \neq 0)\&\&(u_b == 0)$ **then**
    $\theta = arctan(u_a/(u_c - mg))$
    $\phi = 0$
    $T = -(u_c - mg)/cos\theta$
  end if
else
  $\theta = arctan(u_a/(u_c - mg))$
  $\phi = arctan(-u_b/u_a * sin\theta)$
  $T = u_b/sin\phi$
end if

The position subsystem (Outer-loop) controller's output signals ϕ^*, θ^* becomes the desired Euler angles and is sent to the Euler angle subsystem (Middle-loop). The desired yaw angle ψ^* is an external reference signal instead of the control signal generated by outer-loop controller. The thrust T^* is generated by the actuators directly.

The nonlinear mathematical model of the quadrotor is implemented for simulation in MATLAB and Simulink. The simulation is conducted on quadrotor TC2.0 [Corke, 2011] with its parameters listed in Table 6.1. The position control loop has a longer sampling interval $0.1s$ as it has the slowest desired closed-loop response, whereas the angle control loop and angular velocity control loop have the same shorter sampling interval $0.01s$. Nevertheless, in order to make the angular velocity loop (inner-loop) response faster than the Euler angle loop (middle-loop), the inner-loop's $\boldsymbol{R}$ matrix is tuned to be much smaller than that of the outer-loop. For the $\boldsymbol{Q}$ matrix, as we have discussed before, increasing $\boldsymbol{R}$ has the same effect as decreasing $\boldsymbol{Q}$, so we use the same $\boldsymbol{Q}$ for the inner-loop and the middle-loop. Position subsystem (Outer-loop) controller is a PI controller. Constraints are added on the torques only. The torque on each axis is limited between $-15Nm$ to $15Nm$.

The x and y axis reference signals are set to be sinusoidal signals to test the system's capability of tracking fast changing reference signals. Another reason of using sinusoidal signals is we want to test the control system's performance when quadrotor is constantly flying with large Euler angles. In other words, tracking sinusoidal signals can test the control system's performance under varying operating conditions. The z axis is a ramp signal followed by a constant signal.

It can be seen from Figure 7.7 that the quadrotor follows the reference signals with small errors, which shows the DOPC controller's ability of controlling nonlinear systems. In Figure 7.8, it can be seen that the Euler angle ϕ and θ are constantly varying between $-1rad$ to $1rad$. Those simulation results show that DOPC controller can deal with nonlinearity and make the quadrotor fly with large Euler angles. In Figure 7.9, the torque on each axis is limited between $-15Nm$ to $15Nm$. The simulation results also show that the quadrotor with DOPC controllers can chase a moving object with high accuracy.

Figure 7.10 shows the simulation result of TC2.0 tracking a circular path. As we can see, the tracking error only exists at the initial portion of the circular path. Figure 7.11(a) shows that, with resonant controller, the TC2.0's circular path tracking error

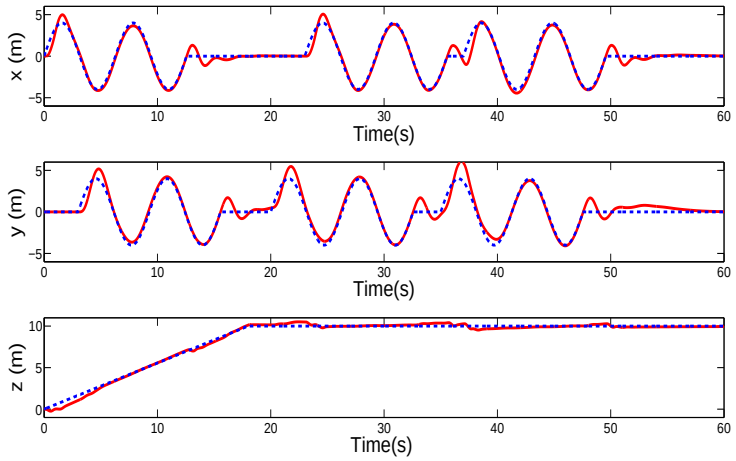


Figure 7.7: quadrotor 3-axis positions. Key: dotted lines reference signals; solid lines output signals

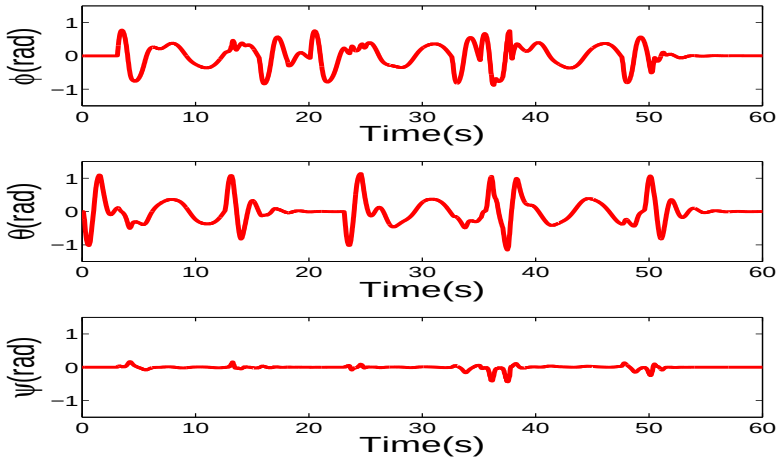


Figure 7.8: quadrotor 3-axis Euler angles

decreases to zero after s, while Figure 7.11(b) shows there is a steady state error when a PI controller is used. Figure 7.12 shows the simulation result of TC2.0 with a resonant controller following a complex trajectory. Tracking error also only exists at the first two seconds and decays to zero afterwards.

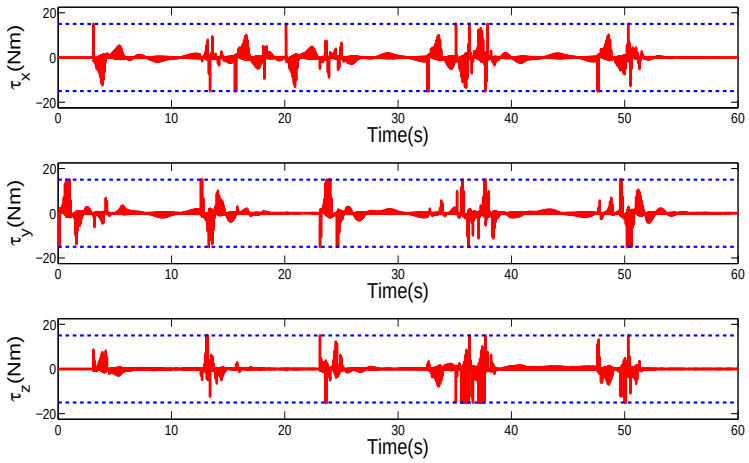
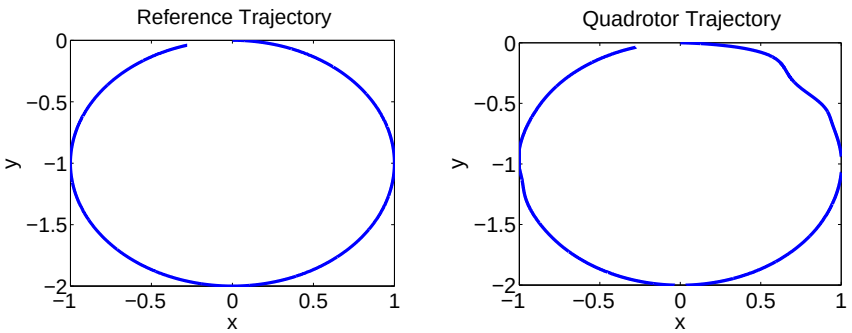
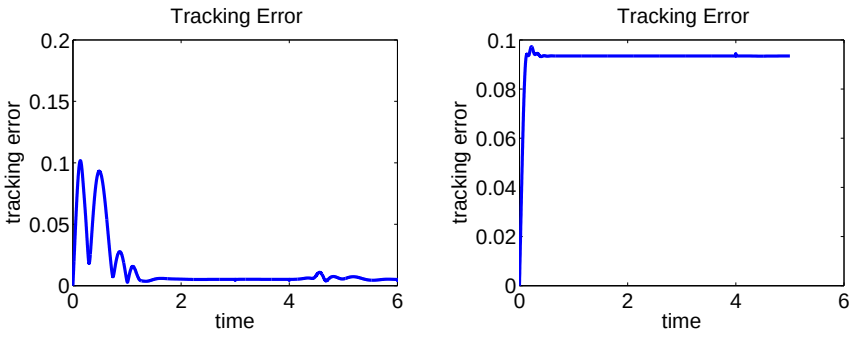


Figure 7.9: quadrotor 3-axis torques with constraints. Key: dotted lines constraints;solid lines torques



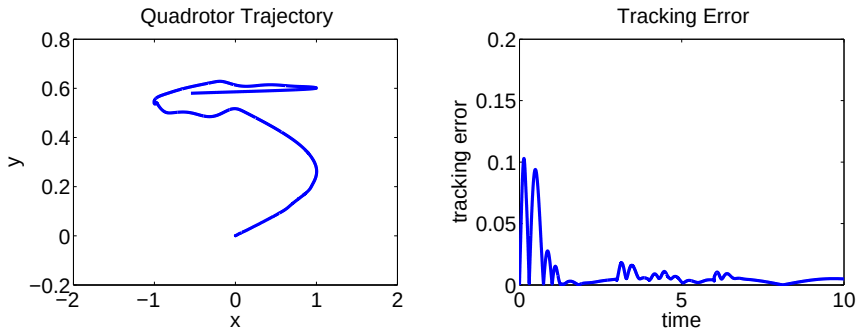
(a) Reference (b) Quadrotor trajectory

Figure 7.10: TC2.0 circular path tracking with resonant controllers



(a) Resonant controller (b) PI controller

Figure 7.11: TC2.0 circular path tracking error comparison



(a) Quadrotor trajectory (b) Tracking error

Figure 7.12: TC2.0 complex trajectory tracking

7.3 Summary

In this chapter an optimization-based nonlinear control technique called DOPC is proposed and applied on various nonlinear plants. It is shown in mathematical derivation and numerical simulations that with the simple design procedures, DOPC controller can deal with nonlinearity and actuator constraints. To design the quadrotor DOPC controller, the quadrotor dynamics is divided into angular velocity, Euler angle and position subsystems. A cascaded control structure is used to stabilize the whole system. DOPC controllers are used in the angular velocity and the Euler angle subsystem to deal with nonlinearity and actuator constraints. Constraints are only applied on the angular velocity subsystem to limit the actual torques. Constraints could be imposed on the angle subsystem to limit the desired angular velocity, but the actual angular velocity might still violate the constraints. For the position subsystem, the nonlinear system is converted to a linear system by defining three new control signals, and the controller is designed based on the linear system. It is shown by the simulation that by using the DOPC controller, the quadrotor can constantly fly with large Euler angles, and thus get the ability of doing faster manoeuvre and working under a wider range of operating conditions.

It should also be pointed that when the weighting matrix $\boldsymbol{R} = 0$, DOPC is similar to feedback linearisation control. However, with the presence of $\boldsymbol{Q}$ and $\boldsymbol{R}$, the closed-loop control performance is modified by tuning $\boldsymbol{Q}$ and $\boldsymbol{R}$, which makes DOPC differ from feedback linearisation. Additionally, DOPC is an optimization based control law and thus it can calculate sub-optimal control signals with constraints. For feedback linearisation control, constraints can only be treated as saturations and thus control signals achieved are not optimal. The application of DOPC on more nonlinear physical systems can be found in Appendix A.

Chapter 8

Conclusions

The objective of this research is to develop a number of systemic approaches to design, implement and assess the quadrotor flight controller. The main contributions of this research comprise three parts: (1)quadrotor PID control auto-tuner design, (2)controller performance assessment, (3)linear and nonlinear quadrotor predictive controller design. This chapter further consolidates the research findings given by the previous chapters and highlights the contributions made by this research.

8.1 Conclusions on the Quadrotor PID Auto-Tuner

In Chapter 2, the quadrotor dynamic model is established, along with the hardware design of two quadrotor test-beds TC1.0 and TC3.0. These two quadrotor test-beds and their physical parameters are used throughout the thesis to compare and validate different control algorithms.

The PID controller, due to its simple structure and robust control performance, is widely used in quadrotor flight control applications. Among various PID design methods, model-based design is often adopted to design the quadrotor controller. So in Chapter 3, the traditional model-based quadrotor PID design method is critically reviewed. The closed-loop frequency response simulations reveal that the commonly used integrator model in quadrotor controller design is over-simplified. On the other hand, using a high order transfer function in model-based design can complicate the design process, and the resultant PID controller may still not perform as expected due to model-mismatching. Those conclusions are also confirmed in the quadrotor

experimental tests. The initial PID parameters calculated based on the quadrotor model fails to stabilise the system, and those PID parameters have to be adjusted by trail and error. The problems existed in the conventional model-based PID controller design become the motivation to develop the PID auto-tuner.

In Chapter 4, a new PID control auto-tuner is developed. The key idea behind the proposed auto-tuner is to fit the dominant dynamics of a physical plant into a simplified integrator plus delay model, which is then used to design the PID controller. The proposed auto-tuner ensures a sound closed-loop control performance, without using a complicated high order model. Besides, the auto-tuner is fully automated and it simplifies the PID control tuning process, and in the meantime it provides control engineers with reliable results. The four steps of the auto-tuner are: relay feedback control, recursive estimation of the frequency response, estimation of the integrating plus delay model and PID controller design. The first three steps are to fit the system's dynamics into a integrator plus delay model (i.e. model identification stage). The last step is to calculate PID parameters based on the identified model (i.e. controller tuning stage). Closed-loop frequency response results in Section 4.3 of Chapter 4 verify that the dominant dynamics of physical systems with different levels of complexity can generally be fitted in to the integrator plus delay model in the auto-tuner's model identification stage. The other group of simulations shows the whole auto-tuner process also applies to high order integrating systems, because of the dominating effect of the integrator in those plants. The auto-tuner is eventually tested on the quadrotors TC1.0 and TC3.0. Both step response tests and flight test results show sound control performance. More importantly, the whole tuning process on TC1.0 and TC3.0 is completed without any human intervention.

Along with the auto-tuner verification experiments, the robustness of various PID configurations are compared. It is shown in Chapter 4 that for both quadrotor inner-loop and outer-loop, the PD controller offers the largest gain margin and phase margin. So the PD controller should be used for the quadrotor inner-loop control. The quadrotor out-loop system, however, needs to use PID controller to ensure zero steady-state error with maintaining large stability margins.

8.2 Conclusions on the Quadrotor Performance Evaluation

Although in Chapter 4 a PID auto-tuner is developed, in some circumstances control engineers still need know the time domain features of a closed-loop system. It is well known that a step response test can reveal whether a closed-loop system can achieve the required settling time, under-damped modes and steady-state error. There are two commonly used means to achieve a system's closed-loop step responses: numerical simulation and step response experiments. Nevertheless, as it is shown in Section 5.4, the step responses achieved from numerical simulations may be inaccurate due to uncertainty of plant physical parameters. Besides, in direct step responses experiments, the system's dynamics can be corrupted by disturbances, so that the time domain features such as rise time, overshoot can hardly be determined. To overcome the drawbacks of numerical simulations and step response experiments, we develop a model identification method to assess the system's step response performance. The first step in the proposed method is data acquisition using relay feedback experiment and the second step is step response identification using frequency sampling filters with generalized least squares method. The proposed model identification method gives a clearer and more intuitive presentation which can be directly used by engineers to evaluate and improve the designed controllers. To validate the proposed identification method, the reference signals recorded in the relay experiments are fed into the identified models. The identified models then generate very similar outputs as what is recorded in the relay tests. Additionally, the reference signals generated in the flying tests are also fed into the identified model. The identified models' outputs are close to the flying test results, but errors exist. It should be emphasized here that the proposed model identification method is not to replace the actual flying test. Its advantages over the numerical simulation and the normal step response test make it a more reliable way to evaluate the designed closed-loop system performance in a safe environment.

8.3 Conclusions on the Quadrotor Preictive Control

Chapter 6 and Chapter 7 cover the topic of linear and nonlinear predictive control of quadrotor. Compared with the PID control, the predictive control, including MPC and DOPC, can optimize the manipulated signals when constraints are present. Additionally, because both MPC and DOPC are MIMO control techniques, the quadrotor controller design process is more straight forward and decoupling of the control signals can be avoided. When a quadrotor at most of the time is required to hover or fly with small Euler angles, liner model predictive control, including cascade and centralized MPC control, can fulfil the control requirements.

In Chapter 6, linear MPC quadrotor control is presented. Even though it is not the first time MPC has been utilized as the quadrotor controller, the thorough comparison of centralized MPC and cascade MPC quadrotor controllers is new. The key findings of this chapter is summarized as below.

- As an optimization-based MIMO control law, both cascade and centralized MPC controllers can fulfil the quadrotor position control requirements and handle actuator constraints.
- The cascade MPC shows better disturbance rejection capability than the centralized MPC. The mean squared errors of cascade MPC are much smaller than the centralized MPC controller in the quadrotor position control simulations.
- The key advantage of cascade MPC over centralized MPC in quadrotor control problem is all the intermediate control signals (Euler angles references in quadrotor position control application), can be limited within a narrow range so that the operational conditions of the linearized MPC controller can be maintained.
- For quadrotor cascade MPC controller, using the quadratic programming to handle both inner-loop and outer-loop constraints provides better results than using pure saturation constraints.
- As the quadrotor's inner-loop and outer-loop dynamics possess different time constants, tuning MPC controllers for a cascade system is much easier than that in the centralized MPC design.

In Chapter 7, a novel optimization based discrete-time nonlinear predictive controller (DOPC) is developed and applied to the quadrotor system as well as several other electromechanical systems. The advantages of DOPC are listed as follows.

- It can deal with the nonlinearity of wide range of MIMO systems with simple design procedures.
- The use of one-step ahead prediction reduces the computational demand.
- Integrators can be included in the DOPC design so that steady-state errors due to step changes in the reference signals and low frequency disturbances can be eliminated.
- Weighting matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are added to the cost function and as a result the closed-loop response speed can be adjusted.
- For the systems with less input signals than their state variables, a cascade DOPC control structure is adopted. By dividing the original plant into several subsystems with equal numbers of inputs and state variables, multiple DOPC controllers can be design accordingly.
- Quadratic programming can be employed to deal with control signal constraints.

The proposed DOPC controller is tested on several plants by numerical simulations. In the quadrotor control case, the use of DOPC enables the quadrotor constantly fly under large Euler angles without degrading the control performance. Thanks to the large Euler angle condition, DOPC angle control can be used along with the resonant position controller to realize accurate tracking of periodic signals.

Several more applications of the DOPC can be found in the appendix of this thesis. In the AUV control application, the optimized torques obtained from the DOPC controller results in the same depth control performance with less pitch variation. In the pendulum control case, DOPC shows superior control performance than the feedback linearisation controller. Additionally, the comparison between DOPC and PI controllers shows DOPC is more robust to unmodelled dynamics.

8.4 Future Research

- In order to perform quadrotor position control, the indoor (external cameras) or outdoor (GPS) localization systems need to be added to the test platforms.
- The PID control auto-tuner can be extended to the quadrotor position control application. However, to conduct the relay test on the position loop, a new quadrotor test-bed is required. That test-bed should be able to move along only one axis with the other two axes locked. Additionally, the test-bed needs to have a negligible friction to ensure the accuracy of the model identification stage.
- As the proposed auto-tuner is a general PID control tuning method, it has the potential to be applied to any integrating systems. In this thesis, simulation results show the PID tuner can work on some high order integrating systems. This conclusion needs to be verified experimentally on the physical systems (e.g. AC motors, fixed-wing aircraft) in addition to the quadrotor.
- The hardware implementation of linear MPC and DOPC need to be performed. For Laguerre MPC, its main advantage is saving computational resource. So in the future, comparisons need to be performed to quantify how much hardware resource can be saved on a specific microprocessor. For both predictive controllers, hardware implementation of quadratic programming to handle constraints can be done in the future.

Appendices

Appendix A

DOPC Applications on nonlinear systems

A.1 Pendulum DOPC Control

The pendulum system equations are as follows [Khalil and Grizzle, 2002]:

$$\dot{x}_1 = x_2 \tag{A.1}$$

$$\dot{x}_2 = -a \sin x_1 - b x_2 + cu \tag{A.2}$$

$$a = \frac{g}{l}, \quad b = \frac{k}{m_r}$$

where x_1 is the pendulum's position, x_2 is the pendulum's angular speed, u is the torque, g is the gravitational constant, l denotes the length of the rod and m_r denotes the mass of the rod. x_1 is the system's output.

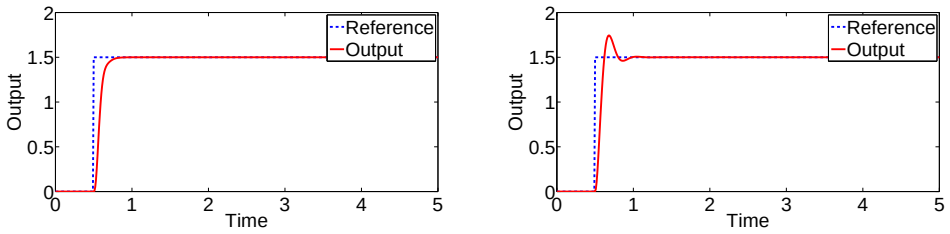
As the cascade control structure is employed, (A.2) is referred as the inner-loop system and (A.1) is referred as outer-loop system. As the inner-loop system has a nonlinear term $-a \sin x_1$, DOPC controller will be used to handle the nonlinearity. However, as the outer-loop dynamics is a linear system, there is no need to use nonlinear controller and thus a PI controller is employed to stabilize the system and eliminate

steady-state errors. The outer-loop system's input is the pendulum's position reference signal, the output signal of the DOPC controller will become the desired angular speed, which becomes the reference signal of the inner-loop system. Then the inner-loop PI controller will generate the desired torque. The design principle of a cascade control system is the the inner close-loop system dynamics needs to be much faster the the outer closed-loop system so that the inner closed-loop system dynamics can be ignored and the outer-loop controller can be designed independently. The inner-loop and outer-loop sampling intervals are $\Delta t_{in} = 1ms$ and $\Delta t_{out} = 10ms$ respectively.

Comparisons are made between cascade DOPC control and feedback linearization control. The latter one is the control law proposed in [Khalil and Grizzle, 2002], where the feedback linearization control signal is given as

$$u = \frac{1}{c}(a \sin x_1 + bx_2 + \ddot{x_1^*} - k_1 e_1 - k_2 e_2)$$
$$e_1 = x_1 - x_1^*, \quad e_2 = x_2 - \dot{x_1^*},$$

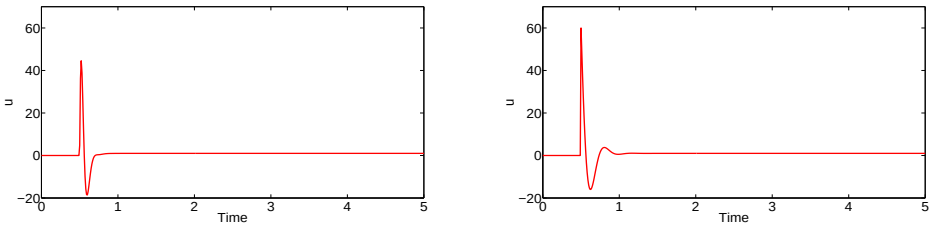
where x_1^* is a reference signal with bounded derivatives $\dot{x_1^*}$ and $\ddot{x_1^*}$, $a = c = 10, b = 1, k_1 = 400$ and $k_2 = 20$.



(a) DOPC controller (b) Feedback linearization controller

Figure A.1: Pendulum position tracking output signals

As we can see from Figure A.1 and Figure A.2, compared with feedback linearization system, cascade DOPC system can achieve almost the same response speed with no overshot and less control signal.



(a) DOPC controller (b) Feedback linearization controller

Figure A.2: Pendulum position tracking control signals

A.2 Autonomous Underwater Vehicle (AUV) Depth and Pitch DOPC Control

The pitch model of the AUV is given as

$$\dot{q_v} = -\frac{1}{I_y}[x_{T_{vf}} T_{vf} + x_{T_{vr}} T_{vr} - z_g W \sin\theta + \frac{1}{2}\rho V^{2/3} C_{D_q} |q_v| q_v]$$

$$q_v = \int_0^t \dot{q_v} dt,\ \theta = \int_0^t q_v dt,$$

where q_v is the angular velocity about the $y-$axis (pitch), I_y is the inertia term for the $y-$axis, T_{vf} is the force generated by the front thruster acting along the $x-$axis, T_{vr} is the force generated by the rear thruster acting along the $x-$axis, $x_{T_{vf}}$ is the distance along x axis between centre of gravity and front vertical thruster, $x_{T_{vr}}$ is the distance along $x-$ axis between centre of gravity and rear vertical thruster, W is the weight of AUV, C_{D_q} is the rotational drag coefficient, z_g is the displacement from the centre of gravity to centre of volume [Steenson et al., 2014].

The AUV's depth model is given as

$$\dot{w_v} = -\frac{1}{m_z}[T_{vf}\cos\theta + T_{vr}\cos\theta + (W - B) - \frac{1}{2}\rho V^{2/3} C_{D_w} |w_v| w_v]$$

$$w_v = \int_0^t \dot{w_v} dt,\ z = \int_0^t w_v dt,$$

where w_v is the velocity in the $z-$direction, B is Buoyancy of AUV, C_{D_w} is the Drag coefficient when the AUV is moving in $z-$axis, m_z is the mass term for the $z-$axis, z is the displacement in $z-$axis.

This is a 2-input, 2-output second order system. The whole system dynamics can be divided into velocity-loop (inner-loop) system, described by (A.3), and position loop (outer-loop) system, described by (A.4). Obviously, the inner-loop dynamics is nonlinear and the outer-loop dynamics is linear. Also the inner-loop control signals T_{vf} and T_{vr} are the forces created by the actuators, so they need to be constrained within the physical limitation. The cascade control structure is used. The DOPC is used as inner-loop controller to deal with nonliearity and constraints. Any linear controller can be used to control the outer-loop and here we use PI controllers in order to eliminate steady-state errors.

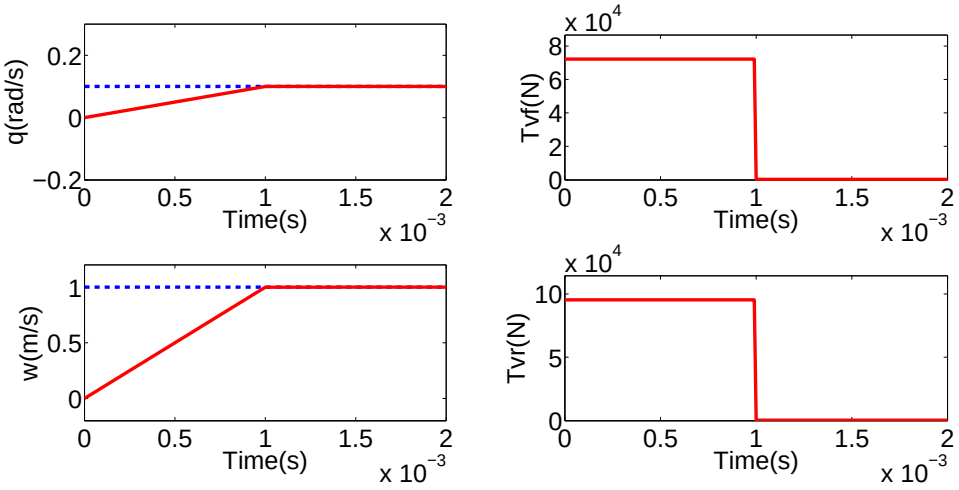
Table A.1: AUV parameters

Parameter	Value	Unit
ρ	1000	kg/m^3
V	0.08	m^3
m_z	167.5	kg
I_y	70	kgm^2
W	540	N
ρ	1000	kg/m^3
$x_{T_{vf}}$	0.55	m
$x_{T_{vr}}$	-0.49	m
z_g	-0.05	m

$$\overbrace{\begin{bmatrix}\dot{q_v}\\ \dot{w_v}\end{bmatrix}}^{\dot{\boldsymbol{x_m}}} = \overbrace{\begin{bmatrix}\frac{z_g W}{I_y}\sin\theta - \frac{1}{2I_y}\rho V^{2/3} C_{D_q}|q_v|q_v \\ \frac{W-B}{m_z} - \frac{1}{2m_z}\rho V^{2/3} C_{D_w}|w_v|w_v\end{bmatrix}}^{\boldsymbol{F(x_m)}} + \overbrace{\begin{bmatrix}-\frac{x_{T_{vf}}}{I_y} & -\frac{x_{T_{vr}}}{I_y} \\ \frac{\cos\theta}{m_z} & \frac{\cos\theta}{m_z}\end{bmatrix}}^{\boldsymbol{B}} \overbrace{\begin{bmatrix}T_{vf}\\ T_{vr}\end{bmatrix}}^{\boldsymbol{u}} \tag{A.3}$$

$$\begin{bmatrix}\dot{\theta}\\ \dot{z}\end{bmatrix} = \begin{bmatrix}q_v\\ w_v\end{bmatrix} \tag{A.4}$$

The first two simulations are AUV inner-loop (velocity-loop) DOPC control. No integrator is used because steady-state errors in the inner-loop system will not affect the



(a) 3-axis angular rates

(b) 3-axis torques

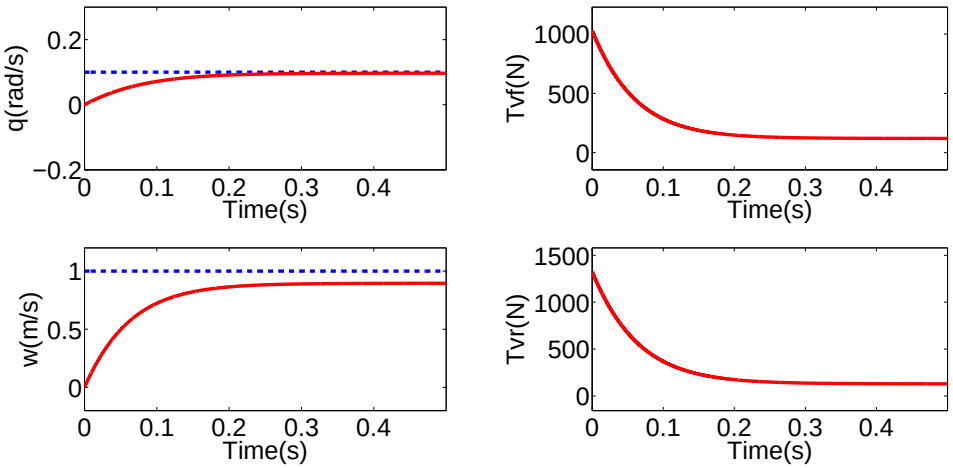
Figure A.3: I-DOPC Quadcopter angular velocity control ($\boldsymbol{R} = 0$). Key: dotted lines reference signals; solid lines output signals or control signals.

outer-loop system's performance much if the inner-loop system's dynamics is faster. Also, as normally in AUV pitch and depth control the pitch angle should be kept small, we assume the pitch angle θ is small so we have $\sin\theta \approx 0$ and $\cos\theta \approx 1$. The sampling interval Δt is $1ms$.

In the first simulation, the weighing matrix $\boldsymbol{Q}$ is an identity matrix and $\boldsymbol{R}$ is a zero matrix, so it can be seen from Figure A.3 the closed-loop dynamics of each channel becomes one sample delay. Also the control signals T_{vf} and T_{vr} are very large. In practice, such large control signals are not available and such fast response is not necessary.

So in the second simulation, a non-zero weighting matrix $\boldsymbol{R} = 5 \times 10^{-9} \times \boldsymbol{I}_{2\times 2}$ is used. As we can see from Figure A.4, the system's response is much slower and the control signals are much smaller. Also, steady-state errors exist as no integrator is used in the DOPC controller.

Another two simulations are conducted to control the AUV's pitch angle and depth. Cascade control structure is used. For the inner-loop DOPC controller, weighting matrices are chosen as $\boldsymbol{Q} = \boldsymbol{I}_{2\times 2}$ and $\boldsymbol{R} = 5 \times 10^{-9} \times \boldsymbol{I}_{2\times 2}$. For the control signals T_{vf}



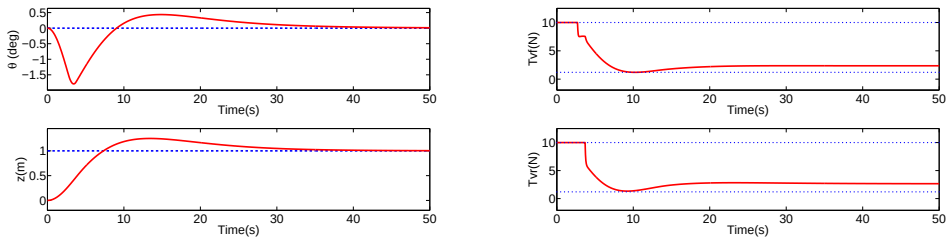
(a) 3-axis angular rates

(b) 3-axis torques

Figure A.4: I-DOPC Quadcopter angular velocity control ($\boldsymbol{R} \neq 0$). Key: dotted lines reference signals; solid lines output signals or control signals.

and T_{vr}, they have the following constants:

$$1.2 \leqslant T_{vf} \leqslant 10$$
$$1.2 \leqslant T_{vr} \leqslant 10.$$



(a) output signals.Key: dotted lines reference signals; solid lines output signals.

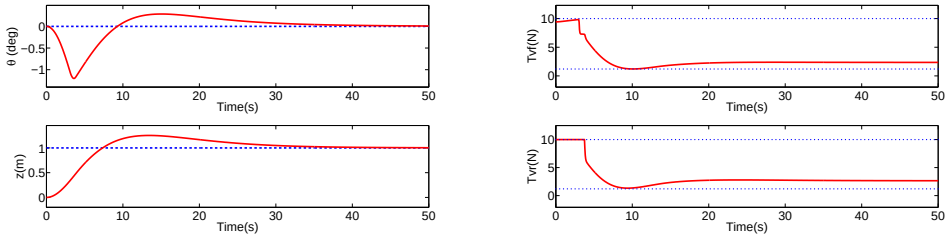
(b) control signals.Key: dotted lines control signal limits; solid lines control signals.

Figure A.5: DOPC AUV pitch angle and depth control (using saturation dealing with constaints).

PI controllers are used as outer-loop controllers and the outer-loop closed system

is tuned to be much slower than the inner closed-loop system.

In the first simulation of AUV pitch angle and depth control, saturations are exerted on the control signals to deal with constants. As it can be seen from Figure A.5(b) that the control signals T_{vf} and T_{vr} are saturated at the initial stage. From Figure A.5(a) it can be seen that the pitch angle varies between -1.6 to 0.5 deg.



(a) output signals.Key: dotted lines reference signals; solid lines output signals.

(b) control signals.Key: dotted lines control signal limits; solid lines control signals.

Figure A.6: DOPC AUV pitch angle and depth control (using quadratic programming dealing with constaints).

In the second simulation of AUV pitch angle and depth control, quadratic programming is used to deal with constants. According to (7.11) and (7.12), we have:

$$E = 10^{-8} \times \begin{bmatrix} 0.5097 & -0.0019 \\ -0.0019 & 0.5085 \end{bmatrix}, \; F = 10^{-5} \times \begin{bmatrix} 0.7857 & -0.5970 \\ -0.7000 & -0.5970 \end{bmatrix}.$$

As we can see from Figure A.6(a), the depth output has similar response as that in the first simulation. However, the pitch angle in this case varies within the range of -1.2 and 0.3 deg, which is much smaller than that in the first simulation. From Figure A.6(b) we can see that by using quadratic programming, even in the initial stage the control signal T_{vf} is less than $10N$.

These simulations results show that by using quadratic programming with DOPC controller, the closed-loop system's performance can be optimized to have more accurate control performance with less control efforts.

A.3 Summary

The simulation results in this section show the superiority of DOPC in various aspects. In the pendulum example, compared with the feedback linearisation controller, the DOPC controller offers an step response with similar settling time, less overshot and less control effort. In the robustness analysis example, the DOPC controller shows more consistent performance than the PI controller subject to model uncertainties. In the AUV control example, the DOPC controller's capability of handling constraints is tested. Compared with the case using pure saturations, DOPC with quadratic programming offers similar AUV depth control performance with less variance on its pitch angle.

Appendix B

TC1.0 and TC3.0 Data Acquisition Original Data and Matlab Code

Figure B.1 is a screen-shot of the data recorded by the TC1.0 data logger.

```
Takecopter v3 attitude PID control
PI+PID Roll kp_in 0.040000 kI_in 0.010000 kp_out 5.000000 ki_out 1.000000 kd_out 0.200000
PI+PID Pitch kp_in 0.040000 kI_in 0.010000 kp_out 5.000000 ki_out 1.000000 kd_out 0.200000
PI Yaw kp 0.400000 ki 0.050000
inner-loop delta t is 0.010000 outer-loop delta t is 0.010000
1Time   2Roll*  3Roll   4Roll_Rate* 5Roll_Rate  6Pitch* 7Pitch  8Pitch_Rate*    9Pitch_Rate 10Yaw_Rate* 11Yaw_Rate
0.0000   -0.3655     -0.1329     1.4930  -0.0011     -0.4090    -0.0100  -1.7999     0.0000  1.3435  0.0000
0.0200   -0.3655     -0.1329     1.4930  0.0000  -0.4090    -0.0095  -1.8119     0.0000  2.0625  0.0011
0.0400   -0.3655     -0.1296     1.4091  -0.0011     -0.4090    -0.0084  -1.8383     0.0011  2.0625  0.0000
0.0600   -0.4090     -0.1264     1.1119  0.0000  -0.4090    -0.0075  -1.8622     0.0011  2.0625  -0.0021
0.0800   -0.4090     -0.1231     1.0304  0.0000  -0.4090    -0.0066  -1.8838     0.0000  2.0625  -0.0011
0.1000   -0.4090     -0.1201     0.9538  0.0000  -0.4090    -0.0057  -1.9054     0.0011  2.0625  0.0000
0.1200   -0.4090     -0.1170     0.8771  -0.0011     -0.4090    -0.0049  -1.9269     0.0011  2.0625  -0.0021
0.1400   -0.4090     -0.1139     0.8004  -0.0011     -0.4090    -0.0039  -1.9509     0.0011  2.0625  -0.0011
0.1600   -0.4090     -0.1111     0.7285  0.0011  -0.4090    -0.0031  -1.9725     0.0011  2.0625  0.0128
0.1800   -0.4090     -0.1082     0.6566  0.0043  -0.4090    -0.0023  -1.9917     -0.0021     2.0625  0.0011
0.2000   -0.4090     -0.1054     0.5871  -0.0043     -0.4090    -0.0014  -2.0000     -0.0032     2.0625  0.0000
0.2200   -0.4090     -0.1027     0.5200  -0.0064     -0.4090    -0.0007  -2.0000     0.0000  2.0625  0.0053
0.2400   -0.4090     -0.1005     0.4648  -0.0075     -0.3631    -0.0002  -1.8145     0.0021  2.0625  -0.0224
0.2600   -0.4090     -0.0976     0.3929  0.0117  -0.2550    0.0005   -1.2896     -0.0043     2.0625  0.0043
0.2800   -0.3375     -0.0951     0.6863  0.0000  -0.0468    0.0012   -0.2655     0.0075  2.0625  -0.0021
0.3000   -0.0551     -0.0926     2.0000  -0.0085     0.1016 0.0020   0.4585  -0.0138     1.8280  0.0468
0.3200   0.1300  -0.0902     2.0000  -0.0628     0.1016 0.0023   -0.3611     -0.0234     0.7608  -0.0106
```

Figure B.1: TC1.0 flight data file: TC-data.txt

The following Matlab script is to decode and plot the flight data.

```
% ====================================
```

```
% FILE            : pictures.m
% DESCRIPTION     : Matlab script to decode and
%                   plot the TC1.0 and TC3.0 flight data.
%
% Author          : Xi Chen
% Created         : 29102014
% ====================================
clear all;
close all;
flightData = importdata('TC-data.txt', '\t', 7);
data=flightData.data;
xLowerLimit=0;
%xUppterLimit=max(data(:,1));
xUppterLimit=50;
% ====================================
% Roll angle
% ====================================
figure(1)
plot(data(:,1),data(:,2),'--b',...
data(:,1),data(:,3),'-r','LineWidth',2)
ylim([-pi/3 pi/3])
xlim([xLowerLimit xUppterLimit])
xlabel('Time','FontSize',40)
%tex_phi = texlabel('phi(rad)');
ylabel('Roll angle','FontSize',40)
set(gca,'FontSize',40)
legend('Reference', 'Output')
% ====================================
% Roll rate
% ====================================
figure(2)
plot(data(:,1),data(:,4),'--b',...
data(:,1),data(:,5),'-r','LineWidth',2)
```

```
ylim([-2 2])
xlim([xLowerLimit xUppterLimit])
xlabel('Time','FontSize',40)
%tex_phi = texlabel('\(rad))');
ylabel('Roll rate','FontSize',40)
set(gca,'FontSize',40)
legend('Reference', 'Output')
% ================================
% Pitch angle
% ================================
figure(3)
plot(data(:,1),data(:,6),'--b',...
data(:,1),data(:,7),'-r','LineWidth',2)
ylim([-pi/3 pi/3])
xlim([xLowerLimit xUppterLimit])
xlabel('Time','FontSize',40)
%tex_phi = texlabel('\(rad))');
ylabel('Pitch angle','FontSize',40)
set(gca,'FontSize',40)
legend('Reference', 'Output')
% ================================
% Pitch rate
% ================================
figure(4)
plot(data(:,1),data(:,8),'--b',...
data(:,1),data(:,9),'-r','LineWidth',2)
ylim([-2 2])
xlim([xLowerLimit xUppterLimit])

xlabel('Time','FontSize',40)
%tex_phi = texlabel('\(rad))');
ylabel('Pitch rate','FontSize',40)
set(gca,'FontSize',40)
```

```
legend('Reference', 'Output')

% ==============================
% Yaw rate
% ==============================
figure(5)
plot(data(:,1),data(:,10),'--b',...
data(:,1),data(:,11),'-r','LineWidth',2)
ylim([-2 2])
xlim([xLowerLimit xUppterLimit])

xlabel('Time','FontSize',40)
%tex_phi = texlabel('\(rad))');
ylabel('Yaw rate','FontSize',40)
set(gca,'FontSize',40)
legend('Reference', 'Output')
```

Appendix C

Quadrotor Simulation Setup

The quadrotor open-loop Simulink model is shown in Figure C.1. The key component in this model is the 6Dof equation of motion block as shown in Figure C.2(a). This block implements Euler angle representation of six-degrees-of-freedom equations of motion. The block's output V_e and X_e represents the translational velocity and position in the inertial reference frame I. ϕ, θ and ψ represents the three Euler angles. DCM_{be} is the coordinate transformation matrix from body reference frame $\{B\}$ to $\{I\}$. The other outputs of the block are not used in this simulator. The first input of the block F_{xyz} is the force vector in B. The other input M_{xyz} is the moment vector in $\{B\}$. The block parameters are given in Figure C.2(b).

In Figure C.1, input ports 1 to 4 are four squared rotor speed, which are converted to moments by multiplying with $TC.E$ matrix. Input 5 is the disturbance. The constant block on the left represents the quadrotor's weight force expressed in $\{I\}$, which is transformed in B by multiplying with DCM_{be}.

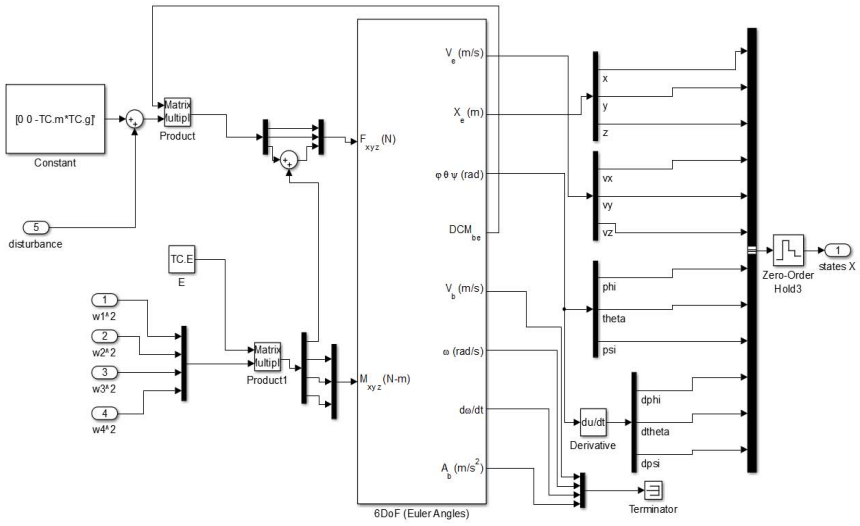
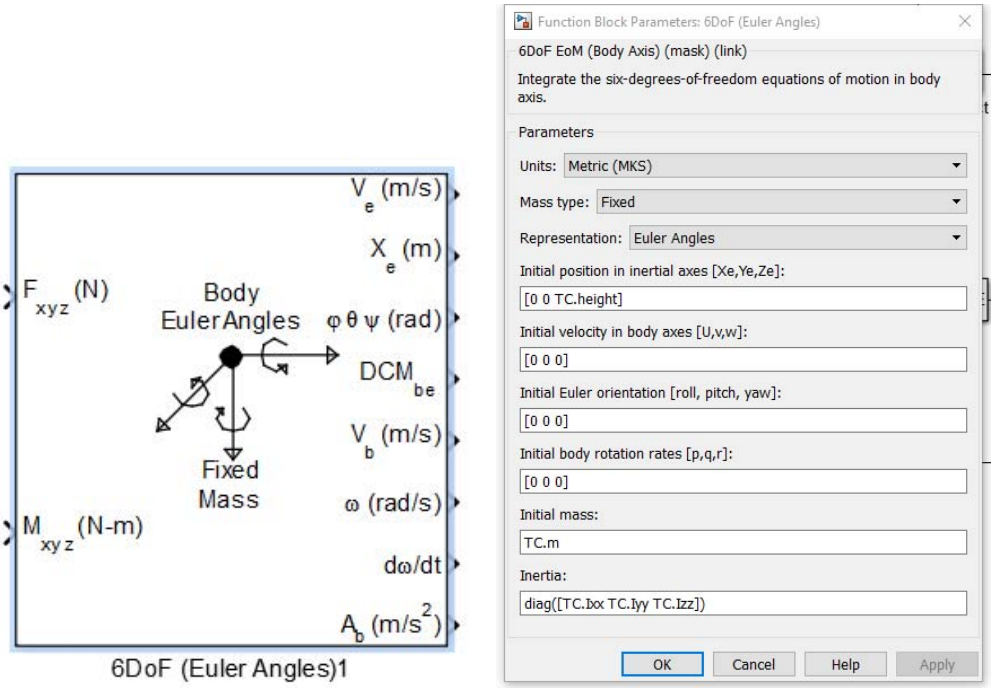


Figure C.1: Quadrotor open-loop simulator in Matlab Simulink



(a) Simulink block (b) Block parameters

Figure C.2: 6 DoF equation of motion block

Appendix D

Implementation of PID in Velocity Form

%%%%%% *PID OBJECT* %%%%%%%%%

```
typedef struct
{
        float kp;
        float ki;
        float kd;
        float tauf;
        float uPast;
        float udPast;
        float yPast;
        float rPast;
        float deltat;
        float lowerLimit;
        float upperLimit;
}PidObject;
```

```
%%%%%% PID FUNCTION %%%%%%%%

float PID(PidObject*pid,float rCurrent,float yCurrent)
{
//pid:  PID object pointer
//rCurrent: the current set point
//yCurrent: the current system output/feedback

float uCurrent; //new control signal
float udCurrent; //new control signal
udCurrent=pid->tauf/(pid->tauf+pid->deltat)*
pid->udPast+pid->kd/(pid->tauf+pid->deltat)*(yCurrent-pid->yPast);

uCurrent=pid->uPast+pid-> kp*
(rCurrent-yCurrent-pid->rPast+pid->yPast)+
(pid->ki*pid->deltat)*(rCurrent-yCurrent)-udCurrent+pid->udPast;
//velocity form PID

if(uCurrent>pid->upperLimit)
uCurrent=pid->upperLimit;
else if (uCurrent<pid->lowerLimit)
uCurrent=pid->lowerLimit;
//output constraints
pid->uPast=uCurrent;
pid->yPast=yCurrent;
pid->rPast=rCurrent;
pid->udPast=udCurrent;
//store the past values
return uCurrent;
};
```

References

K Alexis, G Nikolakopoulos, and A Tzes. Model predictive quadrotor control: attitude, altitude and position experimental studies. *Control Theory & Applications, IET*, 6 (12):1812–1827, 2012. 4

Kostas Alexis, George Nikolakopoulos, and Anthony Tzes. Switching Model Predictive Attitude Control of an Unmanned Quadrotor Helicopter subject to atmospheric disturbances. *Computer Engineering*, (August 2011):1950–1955, 2010. 1

Frank Allgöwer and Alex Zheng. *Nonlinear model predictive control*. Birkhäuser, 2012. 7

B. Armstrong, O. Khatib, and J. Burdick. The explicit dynamic model and inertial parameters of the PUMA 560 arm. *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, 3:510–518, 1986. ISSN 1476-5454. 27

Karl Johan Åström and Tore Hägglund. Automatic tuning of simple regulators with specifications on phase and amplitude margins. *Automatica*, 20(5):645–651, 1984. 5, 6, 7, 12, 48, 88

Karl Johan Åström and Tore Hagglund. *Automatic tuning of PID controllers*. Systems, and Automation Society, 1988. 5, 7

Karl Johan Åström and Tore Hägglund. *Advanced PID Control*. ISA - The Instrumentation, Systems, and Automation Society; Research Triangle Park, NC 27709, 2006. ISBN 978-1-55617-942-6. 5, 7

D. P. Atherton. *Nonlinear Control Engineering-Describing Function Analysis and Design*. Van Nostrand Reinhold Co., London, UK, 1975. 7, 48

Alberto Bemporad and Manfred Morari. Robust model predictive control: A survey. *Robustness in identification and control*, 245:207—-226, 1999. ISSN 13624962. doi: 10.1007/BFb0109870. 7, 8

Marco Bergamasco and Marco Lovera. Identification of linear models for a hovering Quadrotor. *IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY*, 22 (5):161–166, 2014. 1

Qiang Bi, Wen-Jian Cai, Eng-Lock Lee, Qing-Guo Wang, Chang-Chieh Hang, and Yong Zhang. Robust identification of first-order plus dead-time model from step response. 1999. 6

Samir Bouabdallah. Design and control of quadrotors with application to autonomous flying. *Ecole Polytechnique Federale de Lausanne*, 2007. 1, 2

Samir Bouabdallah and Roland Siegwart. Towards intelligent miniature flying robots. In *Field and Service Robotics*, pages 429–440. Springer, 2006. ISBN 3540334521. 4

Samir Bouabdallah, Andre Noth, and Roland Siegwart. PID vs LQ control techniques applied to an indoor micro quadrotor. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 2451–2456. IEEE, 2004. ISBN 0780384636. 1, 18

Y Bouktir, M Haddad, and T Chettibi. Trajectory planning for a quadrotor helicopter. In *Control and Automation, 2008 16th Mediterranean Conference on*, pages 1258–1263, 2008. doi: 10.1109/MED.2008.4602025. 5

E.F Camacho and C. Bordons. *Model Predictive Control*, volume 37. Springer Science & Business Media, 2013. 7

Meric Cetin and Serdar Iplikci. A novel auto-tuning PID control mechanism for nonlinear systems. *ISA transactions*, 58:292–308, 2015. ISSN 1879-2022. doi: 10.1016/j.isatra.2015.05.017. 6

Xi Chen and Liuping Wang. Quadrotor Cascade PID Controller Automatic Tuning. In *Australian Control Conference (AUCC)*, Newcastle, 2016. 5

D. W. Clarke, C. Mohtadi, and P. S. Tuffs. Generalized predictive control Part I.The basic algorithm. *Automatica*, 23(2):137–148, 1987a. 7

D. W. Clarke, C. Mohtadi, and P. S. Tuffs. Generalized Predictive Control Part II Extensions and interpretations. *Automatica*, 23(2):149–160, 1987b. ISSN 00051098. doi: 10.1016/0005-1098(87)90088-4. 7

Peter Corke. *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*, volume 73. Springer, 2011. ISBN 3642201431. 1, 18, 113, 143

C.R. Cutler and B.L Ramaker. Dynamic matrix control- A computer control algorithm. In *joint automatic control conference*, 1980. 7

L Derafa, T Madani, and A Benallegue. Dynamic modelling and experimental identification of four rotors helicopter parameters. In *Industrial Technology, 2006. ICIT 2006. IEEE International Conference on*, pages 1834–1839. IEEE, 2006. ISBN 1424407265. 18, 27

Moritz Diehl, H J Ferreau, and N Haverbeke. Efficient Numerical Methods for Nonlinear MPC and Moving Horizon Estimation Problem Formulation. In *Nonlinear model predictive control*, pages 391–417, Berlin Heidelberg, 2009. Springer. ISBN 978-3-642-01093-4. doi: 10.1007/978-3-642-01094-1_32. 9, 14

Rainer Dittmar, Shabroz Gill, Harpreet Singh, and Mark Darby. Robust optimization-based multi-loop PID controller tuning: A new tool and its industrial application. *Control Engineering Practice*, 20(4):355–370, 2012. ISSN 09670661. doi: 10.1016/j.conengprac.2011.10.011. 6

J Escareño, S Salazar, H Romero, and R Lozano. Trajectory Control of a Quadrotor Subject to 2D Wind Disturbances. *Journal of Intelligent & Robotic Systems*, 70 (1-4):51–63, 2013. 5

R Findeisen and L Imsland. State and output feedback nonlinear model predictive control: An overview. *European journal of control*, 9(2-3):179–195, 2003. ISSN 0947-3580. doi: 10.3166/ejc.9.190-206. 7

B.a. Francis and W.M. Wonham. The internal model principle of control theory. *Automatica*, 12(5):457–465, 1976. ISSN 00051098. doi: 10.1016/0005-1098(76)90006-6. 140

Gene F Franklin, J David Powell, Abbas Emami-Naeini, and J David Powell. *Feedback control of dynamic systems*, volume 3. Addison-Wesley Reading, 1994. 129

C.E. Garcia, D.M. Prett, and M. Morari. Model predictive control: theory and practice-a survey. *Automatica*, 25(3):335–348, 1989. 7

Hugues Garnier and Liuping Wang. *Identification of Continuous-time Models from Sampled Data*. Springer, 2008. ISBN 978-1-84800-160-2. 47

Graham Clifford Goodwin, Stefan F Graebe, and Mario E Salgado. *Control system design*, volume 240. Prentice Hall New Jersey, 2001. 130

Lars Grüne and Jürgen Pannek. *Nonlinear model predictive control*. Springer London, 2011. 7

N Guenard, T Hamel, and R Mahony. A Practical Visual Servo Control for an Unmanned Aerial Vehicle. *Robotics, IEEE Transactions on*, 24(2):331–340, 2008. doi: 10.1109/TRO.2008.916666. 4

Shweta Gupte, Paul Infant Teenu Mohandas, and James M. Conrad. A survey of quadrotor unmanned aerial vehicles. *Conference Proceedings - IEEE SOUTHEASTCON*, 2012. ISSN 07347502. 4

Istvan J. Gyöngy and David W. Clarke. On the automatic tuning and adaptation of PID controllers. *Control Engineering Practice*, 14:149–163, 2006. ISSN 09670661. doi: 10.1016/j.conengprac.2005.01.007. 6

Changsu Ha, Zhiyuan Zuo, Francis B. Choi, and Dongjun Lee. Passivity-based adaptive backstepping control of quadrotor-type UAVs. *Robotics and Autonomous Systems*, 62(9):1305–1315, 2014. ISSN 09218890. 2

T. Hagglund and K. J. Astrom. Method and an apparatus in tuning a PIDregulator, 1985. 5

C C Hang and K K Sin. On-line auto tuning of PID controllers based on the cross-correlation technique. *IEEE Transactions on Industrial Electronics*, 38(6):428–437, 1991. ISSN 02780046. doi: 10.1109/41.107098. 5

C. C. Hang, K. J. Astrom, and Q. G. Wang. Relay feedback auto-tuning of process controllers - A tutorial review. *Journal of Process Control*, 12(1):143–162, 2002. ISSN 09591524. doi: 10.1016/S0959-1524(01)00025-7. 5, 7, 11

Clifford Hildreth. A quadratic programming procedure. *Naval research logistics quarterly*, 4(1):79–85, 1957. 14, 110

W. K. Ho, C. C. Hang, and L. S. Cao. Tuning of pid controllers based on gain and phase margin specifications. *Automatica*, 31:497–502, 1995. 7

WK Ho, TH Lee, and W Xu. The direct Nyquist array design of PID controllers. *IEEE Transactions on Industrial Electronics*, 47(1):175–185, 2000. ISSN 02780046. doi: 10.1109/41.824140. 6

G Hoffmann, D G Rajnarayan, S L Waslander, D Dostal, Jang Jung Soon, and C J Tomlin. The Stanford testbed of autonomous rotorcraft for multi agent control (STARMAC). In *Digital Avionics Systems Conference, 2004. DASC 04. The 23rd*, volume 2, pages 12.E.4–121–10 Vol.2, 2004. doi: 10.1109/DASC.2004.1390847. 4

Gabriel M Hoffmann, Steven L Waslander, and Claire J Tomlin. Quadrotor helicopter trajectory tracking control. In *AIAA Guidance, Navigation and Control Conference and Exhibit, Honolulu, Hawaii*, pages 1–14, 2008. 4

Hsiao-Ping Huang, Ming-Wei Lee, and Cheng-Liang Chen. A System of Procedures for Identification of Simple Models Using Transient Step Response. *Industrial & Engineering Chemistry Research*, 40(8):1903–1915, 2001. ISSN 0888-5885. doi: 10.1021/ie0005001. 6

Jyh-Cheng Jeng. Simultaneous closed-loop tuning of cascade controllers based directly on set-point step-response data. *Journal of Process Control*, 24(5):652–662, 2014. ISSN 09591524. doi: 10.1016/j.jprocont.2014.03.007. 5

Jyh-Cheng Jeng and Ming-Wei Lee. Simultaneous automatic tuning of cascade control systems from closed-loop step response data. *Journal of Process Control*, 22(6): 1020–1033, 2012. ISSN 09591524. doi: 10.1016/j.jprocont.2012.04.010. 5

Michael A. Johnson and Mohammad H. Moradi. *PID Control: New Identification and Design Methods*. Springer Verlag, New York, USA, 2005. ISBN 9781852337025. 5, 7, 11

Ibrahim Kaya. Obtaining controller parameters for a new PI-PD Smith predictor using autotuning. *Journal of Process Control*, 2003. 5

Hassan K Khalil and J W Grizzle. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, 2002. 14, 155, 156

Erwin Kreyszig and Edward J Norminton. *Advanced engineering mathematics*, volume XXXIII. 2014. ISBN 9780874216561. doi: 10.1007/s13398-014-0173-7.2. 49

Osman Kükrer. Deadbeat control of a three-phase inverter with an output LC filter. *IEEE Transactions on Power Electronics*, 11(1):16–23, 1996. ISSN 08858993. doi: 10.1109/63.484412. 9

Wook Hyun Kwon and Soo Hee Han. *Receding horizon control: model predictive control for state models*. Springer Science & Business Media, 2006. 7

Lalo Magni, Davide Martino Raimondo, and Frank Allgöwer. *Nonlinear Model Predictive Control*. Springer, New York, 2009. ISBN 9781447123422. doi: 10.1002/oca.4660060112. 9

Daewon Lee, H Jin Kim, and Shankar Sastry. Feedback linearization vs. adaptive sliding mode control for a quadrotor helicopter. *International Journal of Control, Automation and Systems*, 7(3):419–428, 2009. 2, 3

Yibo Li and Shuxi Song. A survey of control algorithms for quadrotor unmanned helicopter. *2012 IEEE 5th International Conference on Advanced Computational Intelligence, ICACI 2012*, pages 365–369, 2012. 1

Yongseob Lim, Ravinder Venugopal, and A. Galip Ulsoy. Auto-tuning and adaptive control of sheet metal forming. *Control Engineering Practice*, 20(2):156–164, 2012. ISSN 09670661. doi: 10.1016/j.conengprac.2011.10.006. 6, 7

Liuping Wang. *Computer Controlled System Design and Implementation*. Springer, 2015. 37, 140

Lennart Ljung. State of the Art in Linear System Identification: Time and Frequency Domain Methods. 2007. 47

Arko Lucieer, Sharon Robinson, Darren Turner, Steve Harwin, and Josh Kelcey. Using a micro-UAV for ultra-high resolution multi-sensor observations of Antarctic moss beds. In *XXII ISPRS Congress*, 2012. 4

Arko Lucieer, Darren Turner, Diana H. King, and Sharon a. Robinson. Using an unmanned aerial vehicle (UAV) to capture micro-topography of antarctic moss beds. *International Journal of Applied Earth Observation and Geoinformation*, 27 (PARTA):53–62, 2014. ISSN 15698432. 4

M Lungu and R Lungu. Adaptive backstepping flight control for a mini-UAV. *International Journal of Adaptive Control and Signal Processing*, 27:635–650, 2013. doi: 10.1002/acs.2330. 3

Tarek Madani and Abdelaziz Benallegue. Backstepping Control for a Quadrotor Helicopter. *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (June):3255–3260, 2006. 2

David Q. Mayne. Nonlinear model predictive control: An assessment. *AIChE Symposium Series. American Institute of Chemical Engineers*, 1997. 9

David Q. Mayne, James B. Rawlings, Crhistopher V. Rao, and Pierre O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6): 789–814, 2000. ISSN 00051098. doi: 10.1016/S0005-1098(99)00214-9. 7, 8

Edward S. Meadows and James B. Rawlings. Model predictive control. In *Nonlinear process control*. Prentice-Hall, Inc., 1997. 9

L Meier, P Tanskanen, F Fraundorfer, and M Pollefeys. PIXHAWK: A system for autonomous flight using onboard computer vision. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2992–2997, 2011. ISBN 1050-4729. doi: 10.1109/ICRA.2011.5980229. 4

A A Mian and Daobo Wang. Nonlinear Flight Control Strategy for an Underactuated Quadrotor Aerial Robot. In *Networking, Sensing and Control, 2008. ICNSC 2008. IEEE International Conference on*, pages 938–942, 2008. doi: 10.1109/ICNSC.2008.4525351. 3

Hanna Michalska and David Q Mayne. Robust receding horizon control of constrained nonlinear systems. *Automatic Control, IEEE Transactions on*, 38(11):1623–1633, 1993. 14

Hyung-Tae Moon, Hyun-Soo Kim, and Myung-Joong Youn. A discrete-time predictive current control for PMSM. *Power Electronics, IEEE Transactions on*, 18(1):464–472, 2003. 9

M. Morari. *Advances in Model-Based Predictive Control*. Oxford University Press, 1994. 8

Manfred Morari and Jay H. Lee. Model predictive control: past, present and future. *Computers & Chemical Engineering*, 23(4-5):667–682, 1999. ISSN 00981354. doi: 10.1016/S0098-1354(98)00301-9. 7

Florent Morel, Xuefang Lin-Shi, J-M Retif, Bruno Allard, and Cyril Buttay. A comparative study of predictive current control schemes for a permanent-magnet synchronous machine drive. *Industrial Electronics, IEEE Transactions on*, 56(7):2715–2728, 2009. 9

P Pounds, R Mahony, and P Corke. Modelling and control of a large quadrotor robot. *Control Engineering Practice*, 18(7):691–699, 2010. 2

D.M. Prett and R.D. Gillette. Optimization and constrained multivariable control of a catalytic cracking unit. In *Joint Automatic Control Conference*, page 73, 1980. 8

S Joe Qin and Thomas a Badgwell. An overview of nonlinear model predictive control applications. *Nonlinear model predictive control*, pages 369–392, 2000. ISSN 09670661. doi: 10.1016/S0967-0661(02)00186-7. 7, 8

S.Joe Qin and Thomas a. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764, 2003. ISSN 09670661. doi: 10.1016/S0967-0661(02)00186-7. 7

Guilherme V. Raffo, Manuel G. Ortega, and Francisco R. Rubio. An integral predictive/nonlinear H control structure for a quadrotor helicopter. *Automatica*, 46(1): 29–39, 2010. ISSN 00051098. 1

James B. Rawlings and Kenneth R. Muske. The Stability of Constrained Receding Horizon Control. *IEEE Transactions on Automatic Control*, 38(10):1512–1516, 1993. ISSN 15582523. doi: 10.1109/9.241565. 8

J. Richalet, J.L. Rault, A., Testud, and J. Papon. Model predictive heuristic control: Applications to industrial processes. *Automatica*, 14(5):413–428, 1978. 7, 8

D.E. Rivera, M. Morari, and S. Skogestad. Internal model control: PID controller design. *Industrial & engineering chemistry process design and development*, 25(1): 252–265, 1986. 7

J Rodriguez, J Pontt, P Cortes, and R Vargas. Predictive Control of a Three-Phase Neutral Point Clamped Inverter. *Power Electron. Spec. Conf. 2005. PESC '05. IEEE 36th*, (4):1364–1369, 2005. ISSN 02759306. doi: 10.1109/PESC.2005.1581807. 9

Jose Rodriguez and Patricio Cortes. *Predictive control of power converters and electrical drives*. John Wiley & Sons, 2012. 9

Julio Ariel Romero, Roberto Sanchis, and Pedro Balaguer. PI and PID auto-tuning procedure based on simplified single parameter optimization. *Journal of Process Control*, 21(6):840–851, 2011. ISSN 09591524. doi: 10.1016/j.jprocont.2011.04.003. 6

J. A. Rossiter and B. Kouvaritakis. Constrained stable generalised predictive control. *IEE Proceedings D (Control Theory and Applications)*, 140(4), 1993. 8

JA Rossiter. *Model-based predictive control: a practical approach*. CRC press, 2013. 7

Inkyu Sa and Peter Corke. Estimation and control for an open-source quadcopter. In *Proceedings of the Australasian Conference on Robotics and Automation 2011*, 2011. 4

Inkyu Sa and Peter Corke. System identification, estimation and control for a cost effective open-source quadcopter. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2202–2209. IEEE, 2012. ISBN 146731403X. 4

Zak Sarris and S T N ATLAS. Survey of UAV applications in civil markets (June 2001). In *The 9 th IEEE Mediterranean Conference on Control and Automation (MED'01)*, 2001. 4

L Singh and J Fuller. Trajectory generation for a UAV in urban terrain, using nonlinear MPC. In *American Control Conference, 2001. Proceedings of the 2001*, volume 3, pages 2301–2308 vol.3, 2001. ISBN 0743-1619. doi: 10.1109/ACC.2001.946095. 4

Sigurd Skogestad. Simple analytic rules for model reduction and PID controller tuning. *Journal of Process control*, 13(4):291–309, 2003. ISSN 03327353. doi: 10.1016/S0959-1524(02)00062-8. 70

Leo V Steenson, Liuping Wang, Alexander B Phillips, Stephen R Turnock, Maaten E Furlong, and Eric Rogers. Experimentally Verified Depth Regulation for AUVs Using Constrained Model Predictive Control. pages 11974–11979, 2014. 157

Céline Teuliere, Laurent Eck, and Eric Marchand. Chasing a moving target from a flying UAV. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 4929–4934. IEEE, 2011. ISBN 1612844545. 4

Bjorn D Tyreus and William L Luyben. Tuning PI controllers for integrator/dead time processes. *Industrial & engineering chemistry research*, 31(11):2625—-2628, 1992. 54

Vadim Utkin, Jürgen Guldner, and Jingxin Shi. *Sliding mode control in electro-mechanical systems*. CRC press, 2009. 14

René Vargas, Ulrich Ammann, and Jose Rodriguez. Predictive approach to increase efficiency and reduce switching losses on matrix converters. *Power Electronics, IEEE Transactions on*, 24(4):894–902, 2009. 9

Antonio Visioli and Qingchang Zhong. *Control of Integral Processes with Dead Time*. Springer, 2011. 54

H Voos. Nonlinear control of a quadrotor micro-UAV using feedback-linearization. In *Mechatronics, 2009. ICM 2009. IEEE International Conference on*, pages 1–6, 2009. doi: 10.1109/ICMECH.2009.4957154. 3

L O Wallace, A Lucieer, and C S Watson. ASSESSING THE FEASIBILITY OF UAV-BASED LIDAR FOR HIGH RESOLUTION FOREST CHANGE DETECTION. In *XXII ISPRS Congress*, 2012. 4

L Wang and WR Cluett. Frequency-sampling filters: An improved model structure for step-response identification. *Automatica*, 1997a. 6

L. Wang and W.R. Cluett. Tuning PID controllers for integrating processes. *IEE Proceedings - Control Theory and Applications*, 144(5):385–392, 1997b. ISSN 1350-2379. doi: 10.1049/ip-cta:19971435. 11, 54

L. Wang, T.J.D. Barnes, and W.R. Cluett. New frequency-domain design method for PID controllers. *IEE Proceedings - Control Theory and Applications*, 142(4):265–271, 1995. ISSN 1350-2379. doi: 10.1049/ip-cta:19951859. 7

Liuping Wang. *Model Predictive Control System Design and Implementation Using MATLAB*. Springer, 2009. 7, 9, 103, 109, 110, 130

Liuping Wang and William R Cluett. *From plant data to process control: ideas for process identification and PID design*, volume 11. Taylor & Francis, 2000. ISBN 0748407014. 7, 11, 50, 54, 55, 89

Liuping Wang and Lu Gan. Integral FCS predictive current control of induction motor drive. In *IFAC 19th World Congress*, 2014. 9

Liuping Wang and Peter Gawthrop. On the estimation of continuous time transfer functions. *International Journal of Control*, 74(9):889–904, 2001. 87

Liuping Wang, M L Desarmo, and William R Cluett. Real-time estimation of process frequency response and step response from relay feedback experiments. *Automatica*, 35(8):1427–1436, 1999. 11, 12, 87

Liuping Wang, Peter Gawthrop, Charlie Chessari, Tony Podsiadly, and Angus Giles. Indirect approach to continuous time system identification of food extruder. *Journal of Process Control*, 14(6):603–615, 2004. 87

Liuping Wang, Peter J. Gawthrop, and Peter C. Young. Continuous time system identification of nonparametric models with constraints. In *the 16th IFAC world congress*, 2005. ISBN 9781604138795. 47

Liuping Wang, Shan Chai, Dae Yoo, Lu Gan, and Ki Ng. *PID and Predictive Control of Electrical Drives and Power Converters*. Wiley-IEEE Press, 2015. 9, 13, 33

Qing-guo Wang, Yong Zhang, and Xin Guo. Robust closed-loop identification with application to auto-tuning. *Journal of Process Control*, 11:519–530, 2001. ISSN 09591524. doi: 10.1016/S0959-1524(00)00030-5. 5, 7

Qing-Guo Wang, Tong Heng Lee, and Chong Lin. *Relay feedback*. 2002. ISBN 9781447111177. doi: 10.1016/j.automatica.2003.09.008. 47

K. David Young, Vadim I. Utkin, and Ümit Özgüner. A control engineer's guide to sliding mode control. *IEEE Transactions on Control Systems Technology*, 7(3):328–342, 1999. ISSN 10636536. doi: 10.1109/87.761053. 14

Peter C. Young. *Recursive Estimation and Time-Series Analysis*. Springer, 2011. ISBN 978-3-642-21980-1. doi: 10.1007/978-3-642-21981-8. 47

Peter C. Young, Hugues Garnier, and Marion Gilson. Simple refined IV methods of closed-loop system identification. In *IFAC*, volume 15, pages 1151–1156, 2009. ISBN 9783902661470. 47

Cheng-Ching Yu. *Autotuning of PID controllers*. 2006. ISBN 978-1-84628-037-5. doi: 10.1016/S0959-1524(00)00006-8. 5

Cheon Yu Jin, Kyung Hwan Ryu, Su Whan Sung, Jietae Lee, and In Beum Lee. PID auto-tuning using new model reduction method and explicit PID tuning rule for a

fractional order plus time delay model. *Journal of Process Control*, 24(1):113–128, 2014. ISSN 09591524. doi: 10.1016/j.jprocont.2013.11.010. 6